1977

A microprocessor controlled automatic data logging system (ADL)

Casko, John David

http://hdl.handle.net/10945/18201

A MICROPROCESSOR CONTROLLED AUTOMATIC DATA LOGGING SYSTEM (ADL)

John David Casko

NAVAL POSTGRADUATE SCHOOL

Monterey, California

THESIS

A MICROPROCESSOR CONTROLLED AUTOMATIC DATA LOGGING SYSTEM (ADL)

by

John David Casko

June 1977

Thesis Advisor David Caswell

Approved for public release; distribution unlimited.

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER	2. GOVT ACCESSION NO.	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) A MICROPROCESSOR CONTROLLED AUTOMATIC DATA LOGGING SYSTEM (ADL)		5. TYPE OF REPORT & PERIOD COVERED Master's Thesis June 1977 6. PERFORMING ORG. REPORT NUMBER
7. AUTHOR(s) John David Casko		8. CONTRACT OR GRANT NUMBER(s)
9. PERFORMING ORGANIZATION NAME AND ADDRESS Naval Postgraduate School Monterey, California 93940		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
11. CONTROLLING OFFICE NAME AND ADDRESS Navla Postgraduate Schaool Monterey, California 93940		12. REPORT DATE June 1977 13. NUMBER OF PAGES 123
14. MONITORING AGENCY NAME & ADDRESS(if different from Controlling Office) Naval Postgraduate School Monterey, California 93940		15. SECURITY CLASS. (of this report) Unclassified 15a. DECLASSIFICATION/DOWNGRADING SCHEDULE

16. DISTRIBUTION STATEMENT (of this Report)

Approved for public release, distribution unlimited.

17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)

18. SUPPLEMENTARY NOTES

19. KEY WORDS (Continue on reverse side if necessary and identify by block number)

microprocessor, digital control, data logging, data acquisition

20. ABSTRACT (Continue on reverse side if necessary and identify by block number)

This paper describes a digital, microprocessor controlled data acquisition system which optimizes man/machine communications. The processor provides digital feedback control, data collection over any number of channels (up to 8), 32 BIT floating point (7 significant digit) mathematics, and

a variety of output formats. The main features of the device are the ability to work in any numerical unit desired by the user, mathematical noise filtering and automatic feedback control.

The particular application under consideration is automatic data collection and angle-of-attack control of a subsonic wind-tunnel. Data are presented to demonstrate the data logging capabilites of the system.

Approved for public release; distribution unlimited.

A MICROPROCESSOR CONTROLLED AUTOMATIC DATA LOGGING SYSTEM (ADL)

by

John David Casko
Lieutenant, United States Navy
B.S., Kansas University, 1970
M.S., University of West Florida, 1972

Submitted in partial fulfillment of the requirements for the degree of

MASTER OF SCIENCE IN AERONAUTICAL ENGINEERING

from the

NAVAL POSTGRADUATE SCHOOL

June 1977

ABSTRACT

This paper describes a digital, microprocessor controlled data acquisition system which optimizes man/machine communications. The processor provides digital feedback control, data collection over any number of channels (up to 8), 32 BIT floating point (7 significant digit) mathematics, and a variety of output formats. The main features of the device are the ability to work directly in any numerical unit desired by the user, mathematical noise filtering and automatic feedback control.

The particular application under consideration is automatic data collection and angle-of-attack control of a subsonic wind-tunnel. Data are presented to demonstrate the data logging capabilities of the system.

TABLE OF CONTENTS

LIST OF FIGURES

ACKNOWLEDGEMENT

The author takes ths opportunity to express his most sincere thanks tc Cdr. David Caswell, USN, thesis advisor, for his logistic support and patience; to Dr. Louis Schmidt, co-advisor for the physical design of the ADL system and wind tunnel interface; to Ted Dunton, chief technician for his tireless and cheerful troubleshooting.

Special thanks go to Charles Lembo, president of CYBERDATA Corporation, Monterey, California for his generous loan of equipment and facilities during a period of hardware malfunction.

Most of all, thanks to my family for their endless support and understanding.

I. INTRODUCTION

A. BACKGROUND

The use of microprocessors (U-P) to control various analog and digital devices has grown exponentially in the past two years. Applications range from TV tennis games and 'smart' traffic lights, to industrial plant monitors and high speed data handling. The state-of-the-art U-P at the time of this writing is the INTEL 8748. This single integrated circuit contains:

1. Central Processing Unit (CPU)
2. 1K bytes of erasable, programmable, read-only memory (EPRCM)
3. 64 bytes of random access memory (RAM)
4. 8-BIT interval timer/event counter
5. clock driver

In addition, this device draws only 150 milliampere (mA) at 5 volts (V).

Microprocessors have greatly enhanced the important technological application called DISTRIBUTED INTELLIGENCE. For example, routine - but time consuming - chores such as parallel to serial data conversion, x-y plotting, equipment polling, etc. can be controlled on site. Engineering analysis of such large interconnected subsystems reduces to a 'black box' problem rather than the more complex problem

of centralized command and control.

B. DISCUSSION

This paper describes the development and construction of an automatic data logging system (ADL) which is configured via software to suit a particular application. The software modification is dynamic in nature, which means that the system operator needs only to type in a few simple commands to change the system input/output (I/O) to measure volts, feet, psi or any other quantity directly without external hardware modification or adjustment. The requirements for the system and an overall description of the ADL hardware are given in chapter II. Chapter III discusses the command words available along with examples of actual output. Chapter IV presents a specific application of the system. Guidelines for interfacing the digital feedback control function with various types of equipment are also given. Chapter V contains the software assembly listing as well as flowcharts and explanations of the more important routines. Chapter VI discusses the use of U-P development systems and gives recommendations for software development. Appendix A is a glossary of U-P and data acquisition terminology which is used throughout the paper.

II. SYSTEM DESCRIPTION

A. REQUIREMENTS

The purposes of a data logging system are twofold. First, the system must be able to take readings from a variety of physical devices. Second, these readings must be converted into a form suitable for data reduction and human interpretation. The obvious use of such a system is taking data over extremely long or extremely short time periods, filtering out noise, controlling external events and providing tabular and/or graphical output. With the above in mind, the following requirements are defined:

1. 8 channels of analog input
2. 1 channel for digital feedback control of some external device.
3. Plain language man/machine interface via serial data transmission.
4. Manual and automatic data acquisition functions.
5. Limited data manipulation.
6. Multiplexed digital voltmeter function.
7. Limited text file storage and editing.

It should be pointed out that the above requirements were defined with the wind-tunnel control function in mind (ch. IV). Nevertheless, the concepts may be extended to other applications with minor software modifications.

B. DEVICE SELECTION

A strictly hardware-oriented implementation of the system requirements was not a valid alternative due to the inherent inflexibility of such designs. Large scale computer installation was prohibitive from cost and under-utilization considerations. It was therefore decided to use an available microprocessor - the INTEL 8008 - to implement all logic and data manipulation functions. This U-P device is the heart of the PROLOG Corporation 805 microprocessor system. Figure 1 is a schematic of the 805 system layout as modified for this project. Appendix B presents vendor specifications for same. Figure 2 shows the overall system layout including the command and communications links between the system components and the operator.

C. INPUT/OUTPUT DEVICES

The man/machine interface was the most difficult task to implement. The major difficulty was not in the physical interface, but the language used for two-way communications. A software driven ASR-33 Teletype was used for command entry, data presentation, and test functions. Although teletype driving wastes CPU time, the time delays involved are still much less than the mechanical time delays of the relays and driving motors which the U-P is controlling.

A group of eight HEWLETT PACKARD 5082-7302 display lights was used to implement the digital voltmeter function. This display is used to set amplifier gains, set nulls and to verify that data present on a particular input are being

processed by the system. The light display is controlled via software to display data in volts.

Up to eight channels of analog data may be multiplexed (DATEL MM-8) into the sample-and-hold unit (DATEL SHM-4), as shown in figure 3. The analog-to-digital (A/D) converter (DATEL ADC-149) has 14-BIT resolution over a 20-volt range. These three devices are also driven via software in order to prcvide various time delays between data samples. The time delays are utilized to mathematically filter out low level noise and A/D glitches from the system. Appendix B contains vendor data for the above mentioned devices.

D. FUNCTIONAL ARCHITECTURE

1. General

The primary advantage of a software configured system is that its processing functions and I/O can be modified wihout external hardware adjustment. This implies that the system possesses a 'general purpose' quality. However, a compromise must be made between a completely general system and one which can be easily implemented by an operator who has little knowledge of the operating system (OS) software or of the dynamics of the system from which this person is collecting data. In order for the data logging system to be used effectively by students in a variety of engineering disciplines, the OS was set up to optimize man/machine interaction. Thus, the operator has no control over such parameters as relay and drive motor transportation lag. These particular system parameters are fixed (see chapter IV) but still provide wide applications such as probe placement and angle setting. Although obvious,

it is worth mentioning that the feedback controlled movement of an external device may not be coincident with data acquisition and reduction, as the U-P can perform only one function at a time. In general, the internal data handling of the microprocessor is transparent to the user.

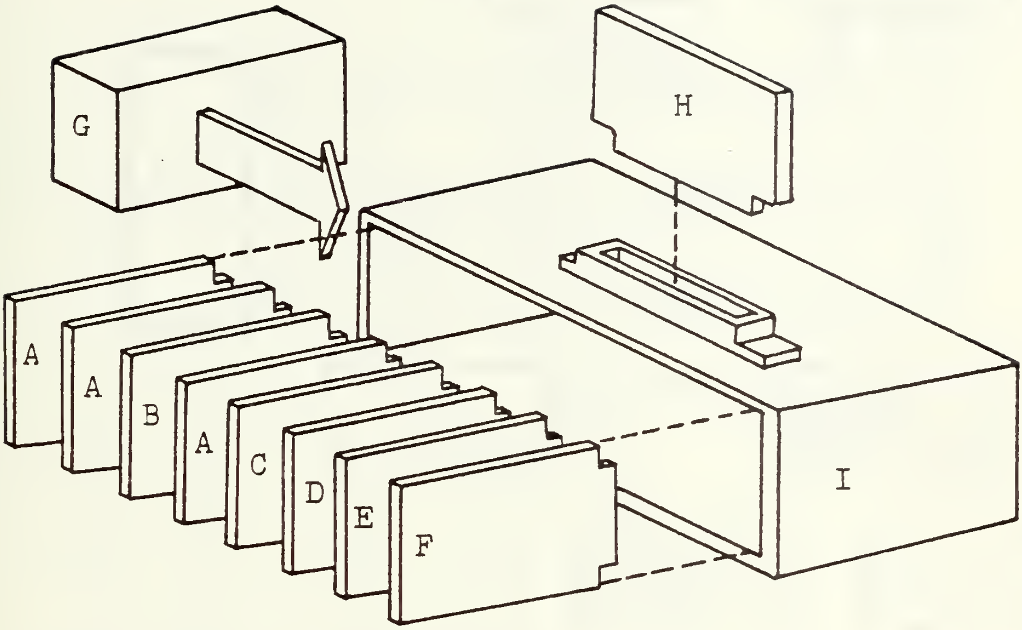
2. Internal

Figure 4 is a flowchart of the basic numerical data conversion processes. Note that two levels of conversion take place. The first level converts data from the 14-BIT binary provided by the A/D converter into a numerical voltage between -10 and +10 volts. This interpolation routine is called before any raw data are processed. The next level of conversion is accomplished with a scaling routine which changes voltage units into any unit desired by the user. If the user does not specify a particular scaling factor, the system defaults to volts for all I/O presentations. Scaling factors can be changed at any time, on any of the input channels; different channels may have different scaling factors.

The mathematics package used is from the INTEL Users Library and is discussed in Appendix C. Although the math package performs all operations with 7 significant digits, numerical output is rounded to 4 significant digits (with choice of decimal or exponential notation). This was done to improve readability of tabulated output and to permit all eight channels to be printed in the limited space provided by the teletype. The only exception is the DUMP routine (ch. III), which always outputs 7 digits. This is because scaling factors of up to 7 digits can be entered by the operator (also fig. 4).

The decimal format presents data between 0.0001 and

9999. The exponential format presents 4 significant digits between 1.000 E-28 and 10.00 E+27. All numerical entries by the operator can be in either format , with the exception of channel numbers, which are only single digit integers.



A - 2K EPROM
B - CPU
C - 4K RAM
D - Input ports
E - Output ports
F - Serial interface
G - Power supply
H - Sockets
I - Card cage

Modular construction of the U-P components enhances expansion.

Figure 1 - PROLOG 805 SYSTEM LAYOUT

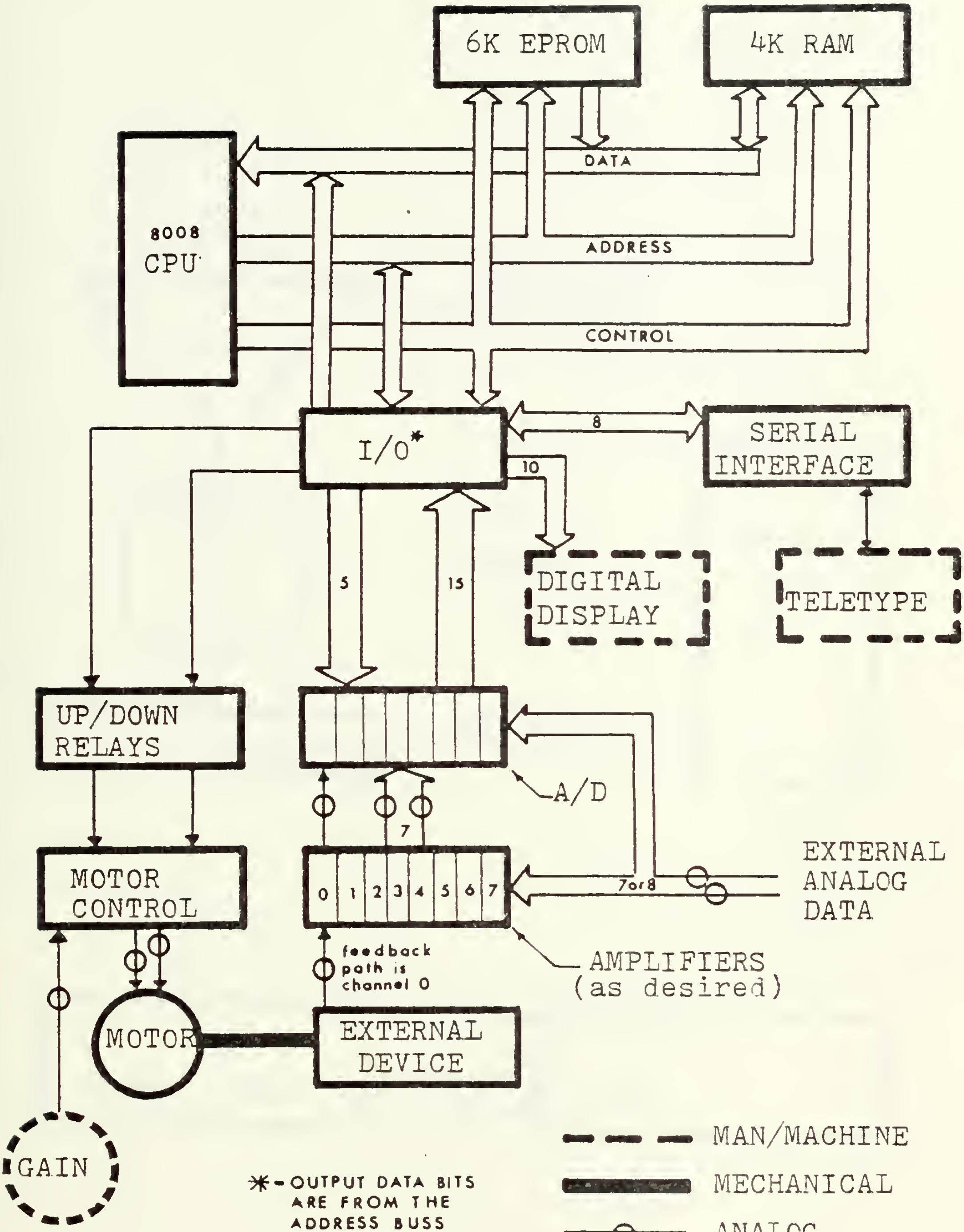


Figure 2 - ADL/805 INTERFACE

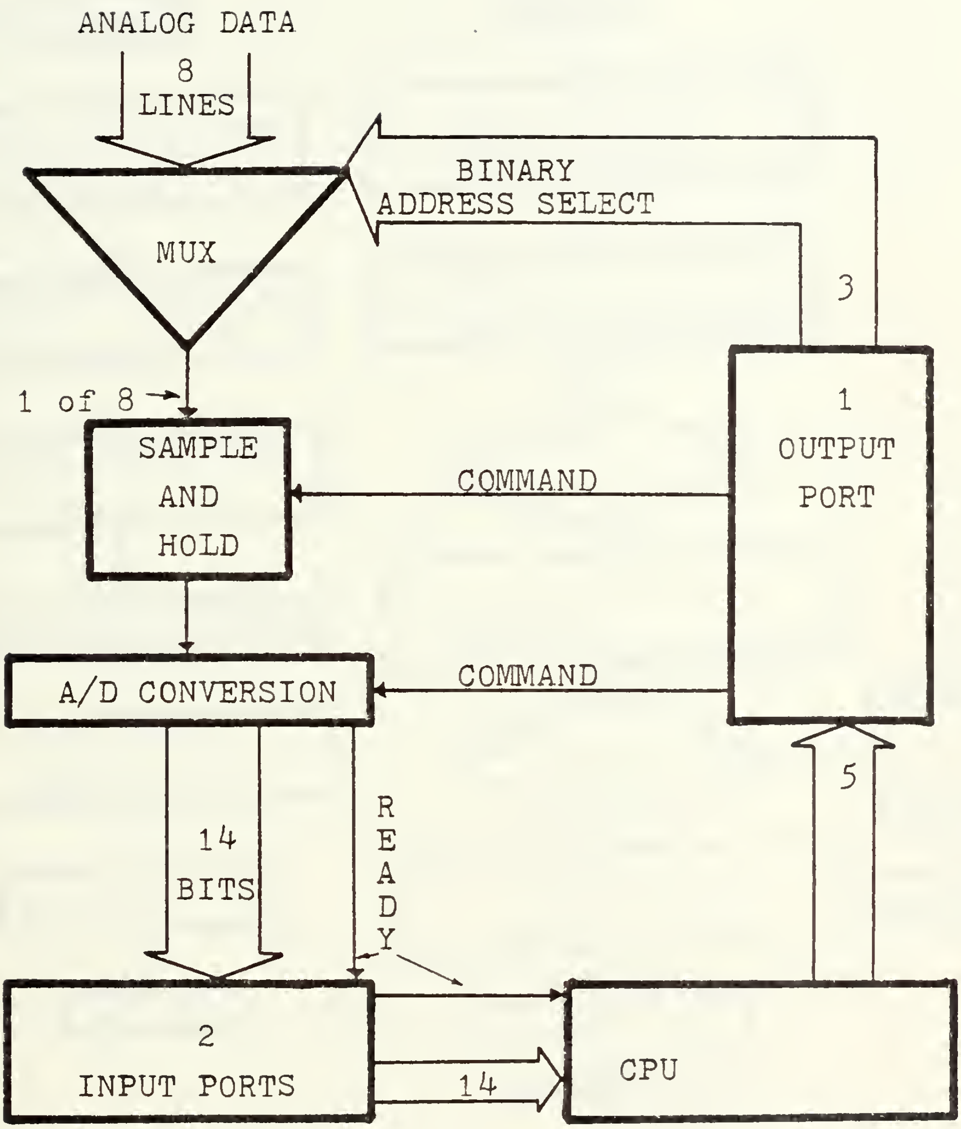


Figure 3 - A/D CONVERSION SYSTEM

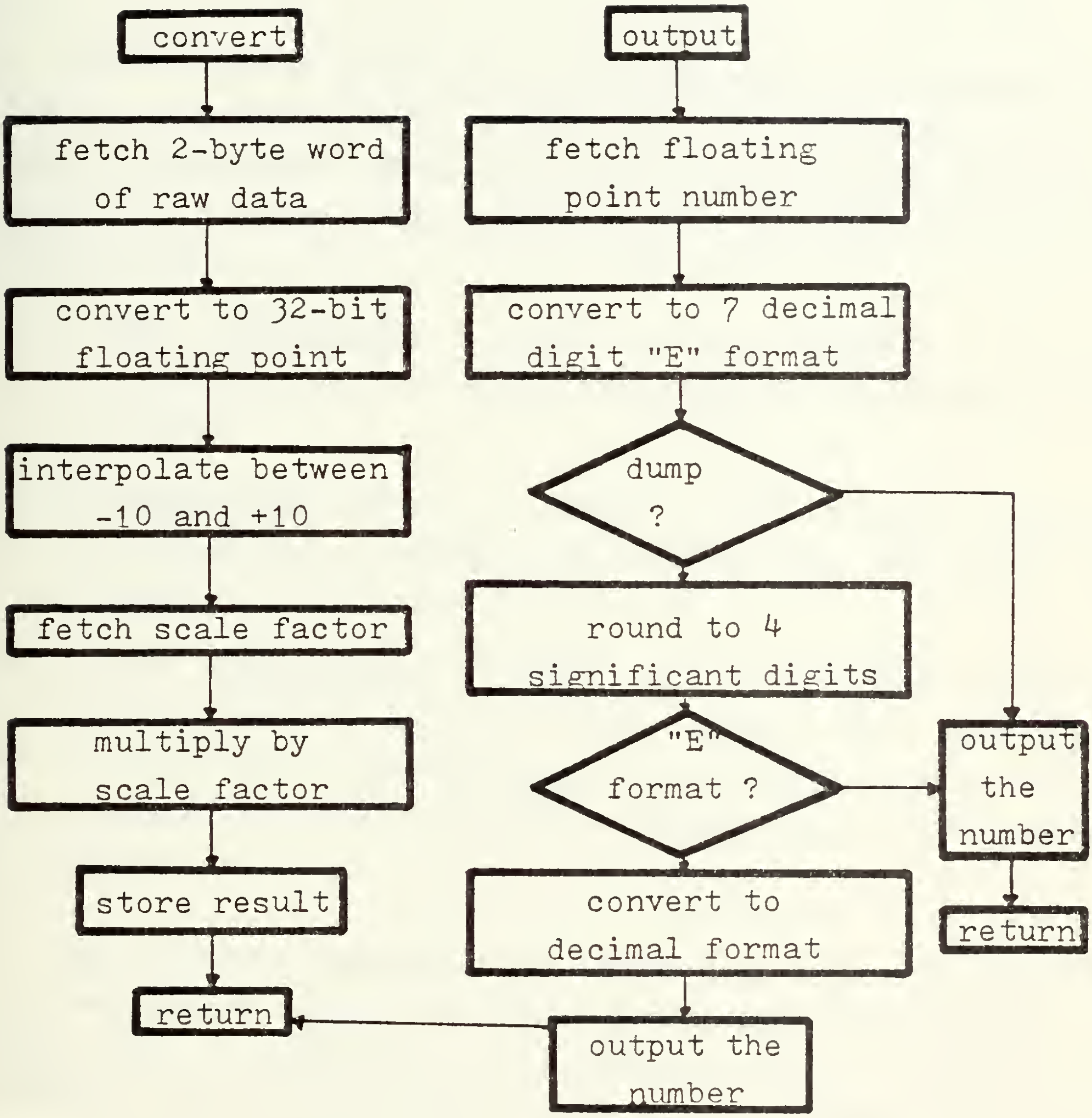

convert
fetch 2-byte word of raw data
convert to 32-bit floating point
interpolate between -10 and +10
fetch scale factor
multiply by scale factor
store result
return
output
fetch floating point number
convert to 7 decimal digit "E" format
dump ?
round to 4 significant digits
"E" format ?
convert to decimal format
output the number
output the number
return

Figure 4 - NUMERICAL DATA CONVERSION METHODS

III. COMMAND WORDS

This chapter defines the commands which the operator may use to communicate with the ADL. A brief explanation of the word is presented; then the rules for it's usage and the ADL response is given. The commands are organized into three categories:

1. DATA DEFINITION: Used to set scaling and delay parameters, and to store the sequence of channels to be scanned.
2. ACQUISITION: These commands start various types of acquisition processing, including the voltmeter and feedback control routines.
3. FILE MAINTENANCE: These are convenience features which provide simple text editing and printing of repetitious table headings. They also provide abort capability and correction capability for misspelled words.

Note that command words are entered in upper case just as they appear in the following paragraphs. When the ADL is ready for a command, it prompts with the symbol >. The command word is then entered followed by pressing the RETURN key. In all examples given, the ADL-generated text is shown in parentheses for illustration purposes only. Section D of this chapter presents copies of actual printout from typical runs in order to further clarify the use of the commands.

A. DATA DEFINITION

1. UNIT

This command is normally the first used. It enables the user to specify a scaling factor so that all data I/O will be in any desired unit. A typical sequence of commands could appear as follows:

```
(>) UNIT
(CHANNEL =) 1
(UNIT/VOLT) 10.5
(CHANNEL =) 6
(UNIT/VOLT) .2
(CHANNEL =)
(>)
```

Thus the user has specified a scaling factor of 10.5 for channel 1 and .2 for channel 6. The abort command was then used to terminate the routine. The ADL responded with > when again ready to accept commands. The result of the above is that, for example, a 3-volt input on all channels will print out as 3.000 on channels 0,2,3,4,5,7; as 31.50 on channel 1 and as .6000 on channel 6.

2. WAIT

this command is used to set a known time delay between sets of data points. The ADL takes 128 data points from each channel and computes an average before a value is printed. Thus, the effects of noise and A/D glitches are

minimized. Because it takes 1 millisecond (ms) for the ADL to set up the multiplexer for a different channel, the WAIT delay must be used with caution in an environment which exhibits periodic noise. If only one channel is scanned, the time between data points will be as entered. However, if all 8 channels are scanned the time between data points on any channel will be the desired time plus 8 ms (1 ms for each channel).

3. SET SCAN

SET SCAN is a dual purpose routine. First of all, if a WAIT command has not been issued it will default to 15 ms and send a message to the operator; otherwise, it proceeds to the next step. The operator is then given the opportunity to specify the type of output format, namely, exponential or decimal. Second, the ADL asks the operator to input the channels to be scanned in the desired sequence. Section D presents some typical examples of SET SCAN usage.

B. ACQUISITION COMMANDS

1. READ

This is the command used to start the voltmeter function. A typical entry would look like:

```
(CHANNEL =) 3
```

When the RETURN key is pressed, the digital display will follow the data on channel 3 (in volts). The display is updated every 10 ms and takes the data through a 43 microsecond window. When the display is in operation, any

noise on the input will show up as a rapidly changing digit. This is unlike most digital voltmeters which integrate the input over a small time interval and present an average reading. This integration process may effectively mask any noise down to fairly low frequencies, depending on the voltmeter being used.

The main use of the above function is to set gain limits on the inputs. To transfer command back to the user, any key on the teletype can be pressed. The ADL responds with > and is then ready for another command.

2. SCAN

SCAN is used to manually control the tabulation of data. Upon command entry, the ADL checks to see if channel assignments have been made via the SET SCAN routine. If the check is negative, a message is sent to the operator and the routine is aborted. If the check is positive, headings are printed out with the proper spacing for the desired numerical format. The ADL then waits for a RETURN at which time a set of data is taken, averaged and printed out with the proper scaling factor applied. The printer carriage is then positioned at the end of the line of data so the user may enter any ccmments. The next set of data is taken when the RETURN key is pressed. Before each line of data is printed, a three digit counter, called the coordination number, is incremented automatically. The SET SCAN routine is used to reset this counter to zero (also automatic). Thus, repeated calls to SCAN or RUN will keep the counter indexing properly. The SCAN routine is terminated by entering the abort command.

3. MOVE

Channel 0 was internally defined as the feedback channel for ADL control of some external device. The operator inputs the desired position (speed, angle, etc.) and the ADL will provide the logic necessary to drive the device to within 4 A/D counts; 1 A/D count is equal to 1.22 millivolt (mV). A sample is taken every 0.8 ms so the maximum slew rate at the input is limited to 6.1 mV/sec in order to guaranty convergence. This routine is used mainly to ensure slew rates are not excessive and that external device mcvement does not exceed acceptable limits. Chapter V presents a detailed flowchart of the feedback logic used in the ADL.

4. RUN

RUN internally calls the SCAN and MOVE routines in repetition in order to automate the tabulaticn of data at many different positions of an external device. A typical data entry sequence for RUN could be:

```
(START POSIT =) 10.0
(STOP POSIT =) 7.5
(INCREMENT  =) .5
```

Note that the start position does not need to be less than the stop position. Also note that the incremental movement is in absolute value. Upon execution (the RETURN key) the ADL prints out column headings as defined by the SET SCAN routine, slews the external device to the start position and starts tabulating data at each of the positions between START and STOP. When the stop position data have been printed out, command is returned to the operator.

C. FILE MAINTENANCE

The following commands are used to input text information and comments.

1. EDIT

When the same heading information will appear as part of the documentation of each run, EDIT is used to enter this information for later use. After the desired text has been entered, the LINE FEED (L/F) key is pressed, followed by the RETURN key. The L/F is needed internally to mark the end of the file; this is the only routine that terminates with other than just the RETURN key. If the L/F key were not used, the entire buffer space (256 characters) would print out. At this time, if the END-OF-FILE symbol were not detected, an error message would be sent to the operator.

The EDIT mode can also be used to change or correct the text at any time. The routine is entered via the command word; the CONTROL and Z keys (cntrl-Z) are pressed simultaneously to step through the file. When the proper place in the text is reached, the RUBOUT key is pressed, then the new character is entered. Note that this entry is not inserted between two characters, but rather it overwrites; any number of characters can be reentered. In order to exit the routine without using the L/F key (which would truncate the text), the cntrl-A keys are used. Section D shows the construction and edit of a file.

2. FILE

The text entered with the EDIT command is printed out upon execution of FILE. Two lines are skipped automatically at the beginning and at the end of the file.

3. DUMP

Execution of DUMP will cause the contents of the conversion factor buffer to be printed on the teletype. This enables the user to verify the scaling factors which are being applied to each channel. Numbers are printed with 7 significant digits.

4. MTEST

This command is executed automatically upon system reset or when power is applied. All the ram area is tested by first writing 00H to each location, reading it back and comparing the result with 00H; the same process is repeated with FFH (Appendix C contains an explanation of hexadecimal notation). If an error were detected, the contents of the bad location would be printed out along with its address. This enables the operator to identify the particular circuit component which has malfunctioned. The routine can also be entered as a command, but use of this function resets all default values just as a system reset.

5. CNTRL-A

The abort command is used to terminate execution of all routines except RUN and READ. When the cntrl-A keys are used, command is transferred back to the operator and the system responds with a >. The RUN routine can only be

terminated with a system reset; the READ routine is terminated by pressing any of the teletype keys.

6. CNTRL-R

Pressing these two keys causes the phrase RUN NO. to be printed. The keyboard is then opened for the insertion of any desired alpha/numeric single line sequence.

7. CNTRL-C

This command causes **** to be printed in order to flag a comment. Note that this is a command routine and can only be entered after a > prompt by the ADL.

8. CNTRL-Z

Pressing these keys causes an internal counter to advance forward through the input buffer. In this manner, the contents of memory can be displayed (see EDIT).

9. RUBOUT

This key is used mainly to correct spelling errors or to correct data entries without aborting a routine. For example, assume the operator wants to enter SCAN but notices that SVAN has been typed by mistake. The RUBOUT key is pressed three times. Each time it is pressed, the previously entered character is printed and an internal counter counts backwards through the input buffer. The operator then retypes the correct letters (CAN) and the correction is complete. The teletype entries would then look like:

SVANNAVCAN

When executed, the ADL will only see SCAN.

Ccmmand word recognition is accomplished by summing the binary codes of each of the letters of the word. The result is then compared with a list of valid sums which are contained in memory (check-sum). Since the result of a summation does not depend on the order of the addends, the letters of the command word may be entered in any order (e.g., SCAN, NACS, etc.). Although this method could lead to ambiguity problems, the vocabulary of the ADL is small enough to prevent such an inconsistency. Any command ,text or data entry can be edited with the RUBOUT key at any time.

D. EXAMPLES

The figures following this section are copies of actual ADL sessions. All ADL-generated messages are underlined the first time they appear for illustration purposes only.

Figure 5 shows a DATA DEFINITION sequence. The UNIT command was used to override the volt default on channels 0,1 and 2. The SET SCAN routine was used to select decimal format and to sequence channels 6,7,0,3,6. Note that the channels do not need to be in any particular sequence and that one channel may be used more than once. The use of a channel more than once enables the user to check thc effectiveness of the noise filtering algorithm in a particular application. In this case, data taken on the first channel 6 scan will be 19 ms out of phase with the second channel 6 scan (15 ms for the delay parameter and a 1 ms intercycle delay for each channel). The operator then used the wait routine to change the wait parameter to 3 ms.

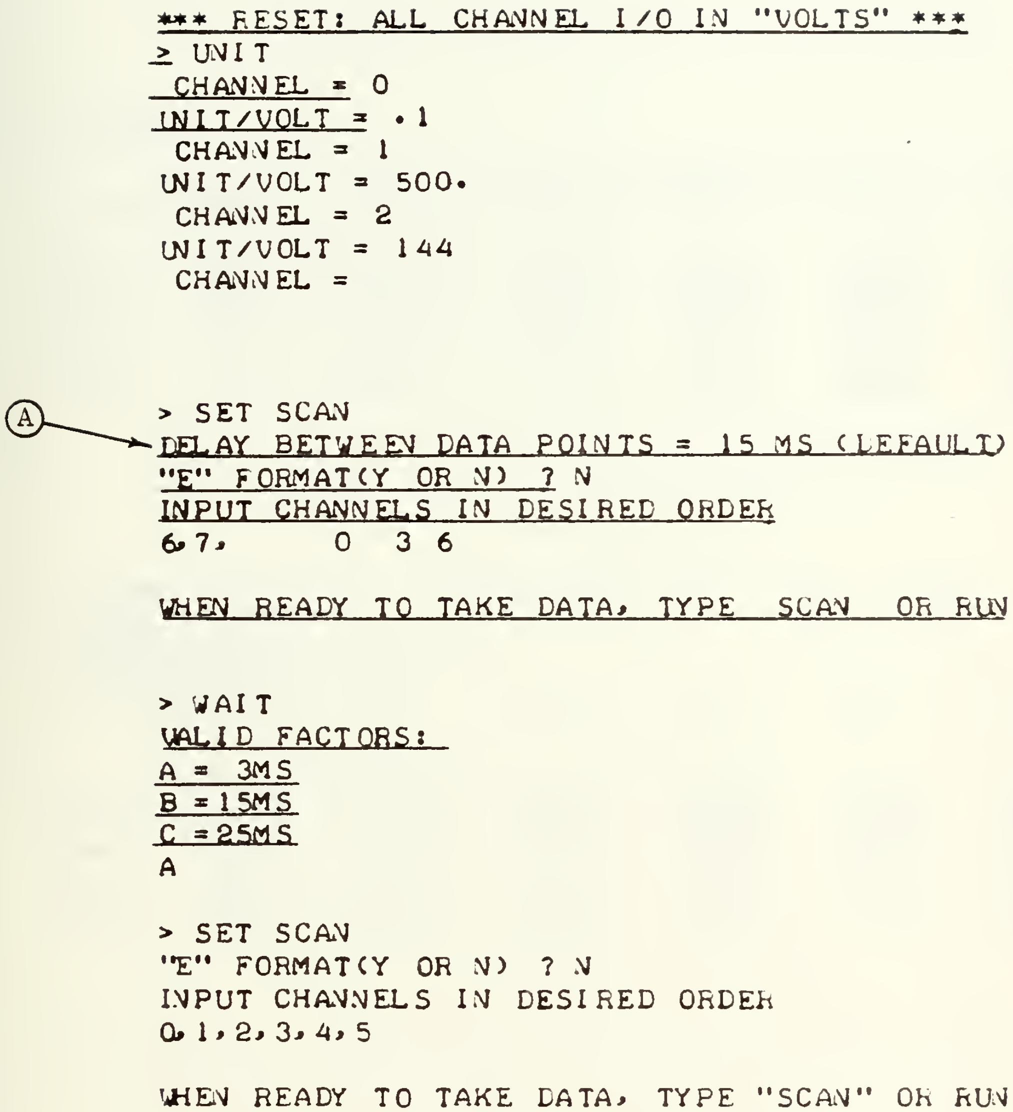
The subsequent call to the SET SCAN routine did not produce the default message (labeled A on the previous call).

Figure 6 shows a typical SCAN sequence which resulted from the commands entered from fig. 5. SCAN was used to take 5 sets of data, then RUN was used to take 5 sets. It is important to note that channel 0 must be included in the SET SCAN definition before RUN is executed. This is because channel 0 is the feedback path for the digital control functions.

Figure 7 shows the ADL response to improper inputs. After a reset, the operator tried to execute SCAN without first defining the channel sequence. The next example in this figure shows an invalid command followed by some examples of using the RUBOUT key to correct various entries.

In fig. 8 the operator did not use a LINE FEED/RETURN sequence to mark the end of the text. The resulting call to FILE is then shown.

Figures 9 and 10 present the data from a wind tunnel calibration session. Figure 11 is a graph of the lift data (channel 2) versus angle-of-attack (AOA, channel 0). Notice that the scaling factors for channels 0 and 2 were selected so that their respective output would be read in degrees and pounds directly. Figure 12 shows a run which utilized exponential format.

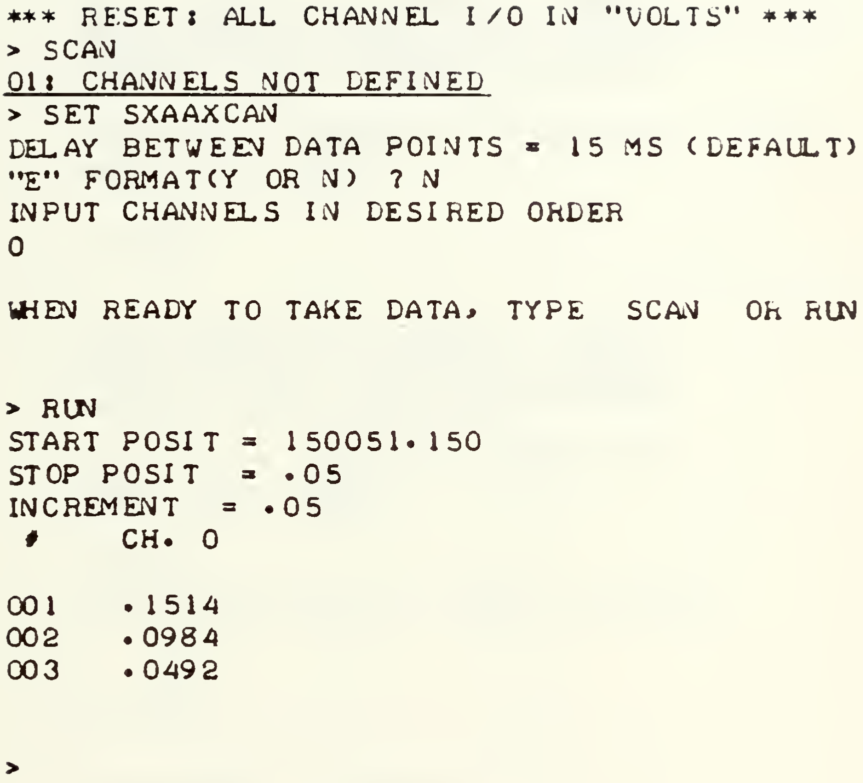


```
*** RESET: ALL CHANNEL I/O IN "VOLTS" ***
> UNIT
 CHANNEL = 0
UNIT/VOLT = .1
 CHANNEL = 1
UNIT/VOLT = 500.
 CHANNEL = 2
UNIT/VOLT = 144
 CHANNEL =



> SET SCAN
DELAY BETWEEN DATA POINTS = 15 MS (DEFAULT)
"E" FORMAT(Y OR N) ? N
INPUT CHANNELS IN DESIRED ORDER
6,7,      0  3  6

WHEN READY TO TAKE DATA, TYPE  SCAN  OR RUN


> WAIT
VALID FACTORS:
A =  3MS
B =15MS
C =25MS
A

> SET SCAN
"E" FORMAT(Y OR N) ? N
INPUT CHANNELS IN DESIRED ORDER
0,1,2,3,4,5

WHEN READY TO TAKE DATA, TYPE "SCAN" OR RUN
```

Figure 5 - DATA DEFINITION EXAMPLES

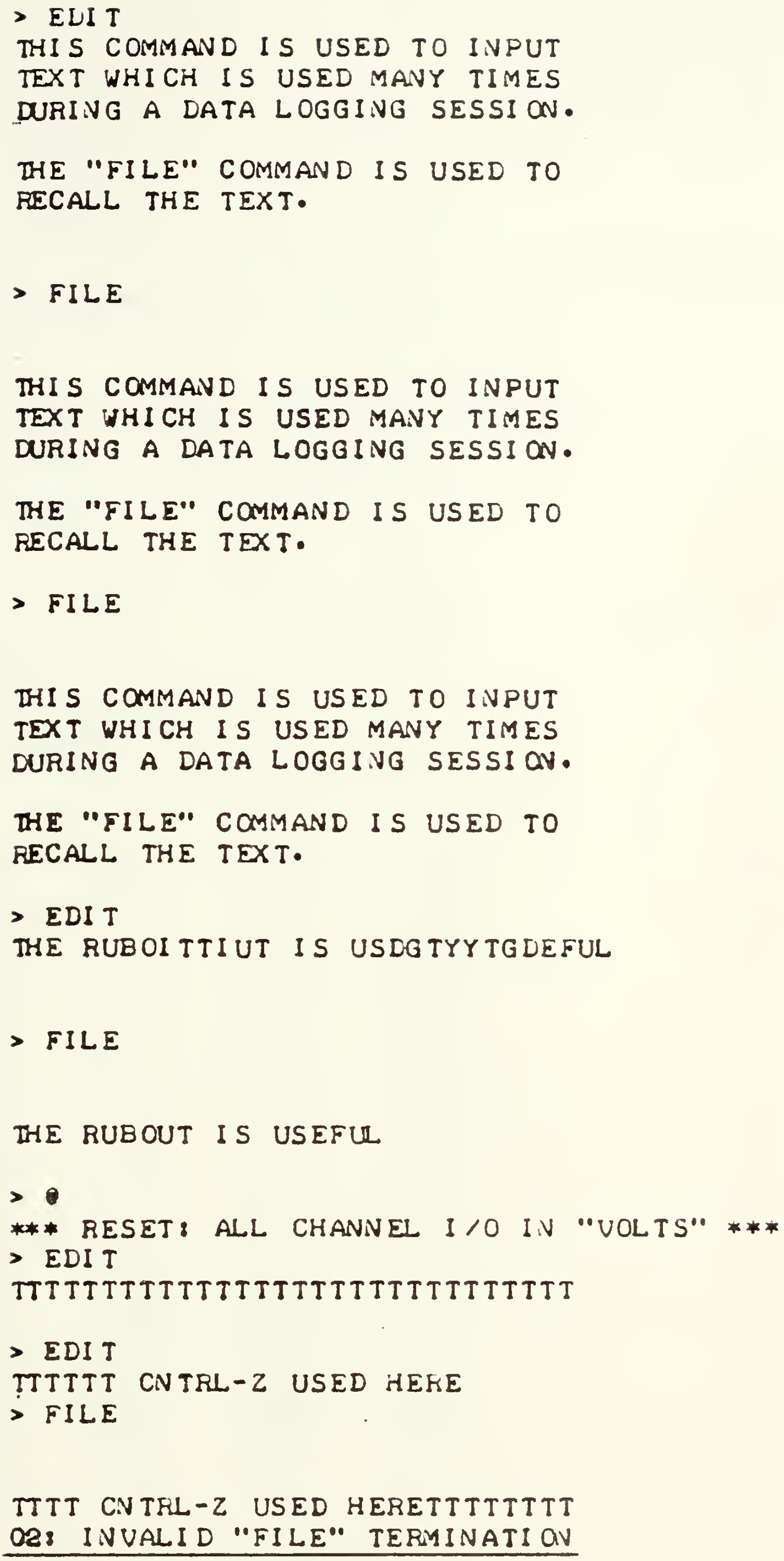
```
> SCAN

  #      CH. 0     CH. 1     CH. 2     CH. 3     CH. 4     CH. 5
_________________________________________________________________

001     .0997    -25.98    -1.308    -.0574     .0355     .0006
002     .0997    -25.95    -1.306    -.0574     .0357     .0006
003     .0997    -26.05    -1.334    -.0572     .0366     .0006
004     .1118    -25.87    -1.276    -.0641     .0424     .0006
005     .1119    -25.89    -1.163    -.0639     .0426     .0006



> RUN
START POSIT =  .1
STOP POSIT  =  -.02
INCREMENT   = .03
  #      CH. 0     CH. 1     CH. 2     CH. 3     CH. 4     CH. 5

006     .0995    -25.86    -1.261    -.0568     .0371     .0006
007     .0698    -25.56    -106.0    -.3309    -.1288     .0006
008     .0398    -25.56     80.72     .1612     .0127     .0006
009     .0097    -25.41     27.19    -.0509    -.0454     .0005
010    -.0202    -25.32    -22.10     .0410     .0162     .0006


>
```

Figure 6 - SCAN AND RUN EXAMPLES



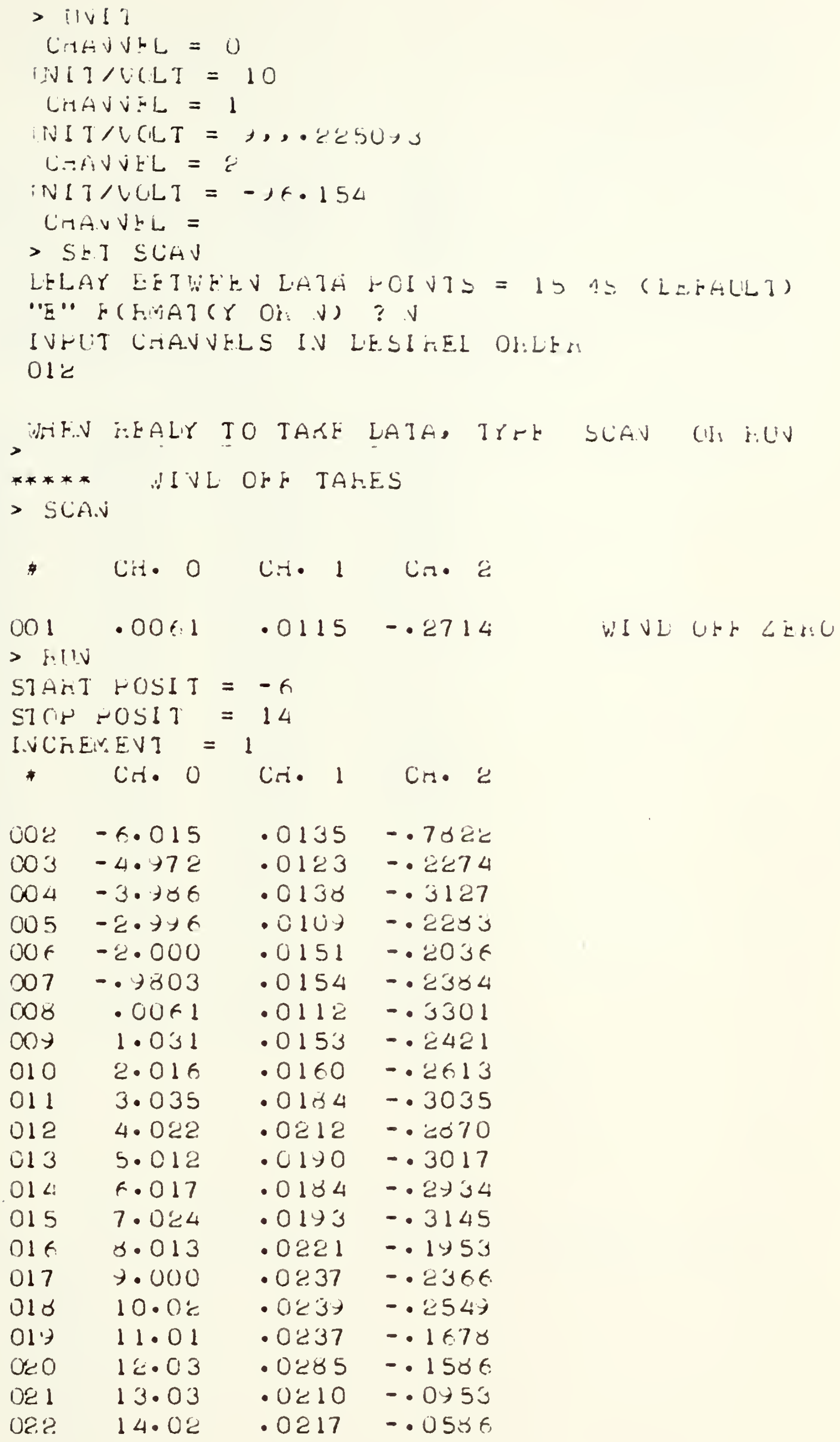
```
*** RESET: ALL CHANNEL I/O IN "VOLTS" ***
> SCAN
01: CHANNELS NOT DEFINED
> SET SXAAXCAN
DELAY BETWEEN DATA POINTS = 15 MS (DEFAULT)
"E" FORMAT(Y OR N) ? N
INPUT CHANNELS IN DESIRED ORDER
0

WHEN READY TO TAKE DATA, TYPE  SCAN  OR RUN


> RUN
START POSIT = 150051.150
STOP POSIT  = .05
INCREMENT   = .05
 #     CH. 0

001    .1514
002    .0984
003    .0492


>
```

Figure 7 - IMPROPER INPUT EXAMPLE



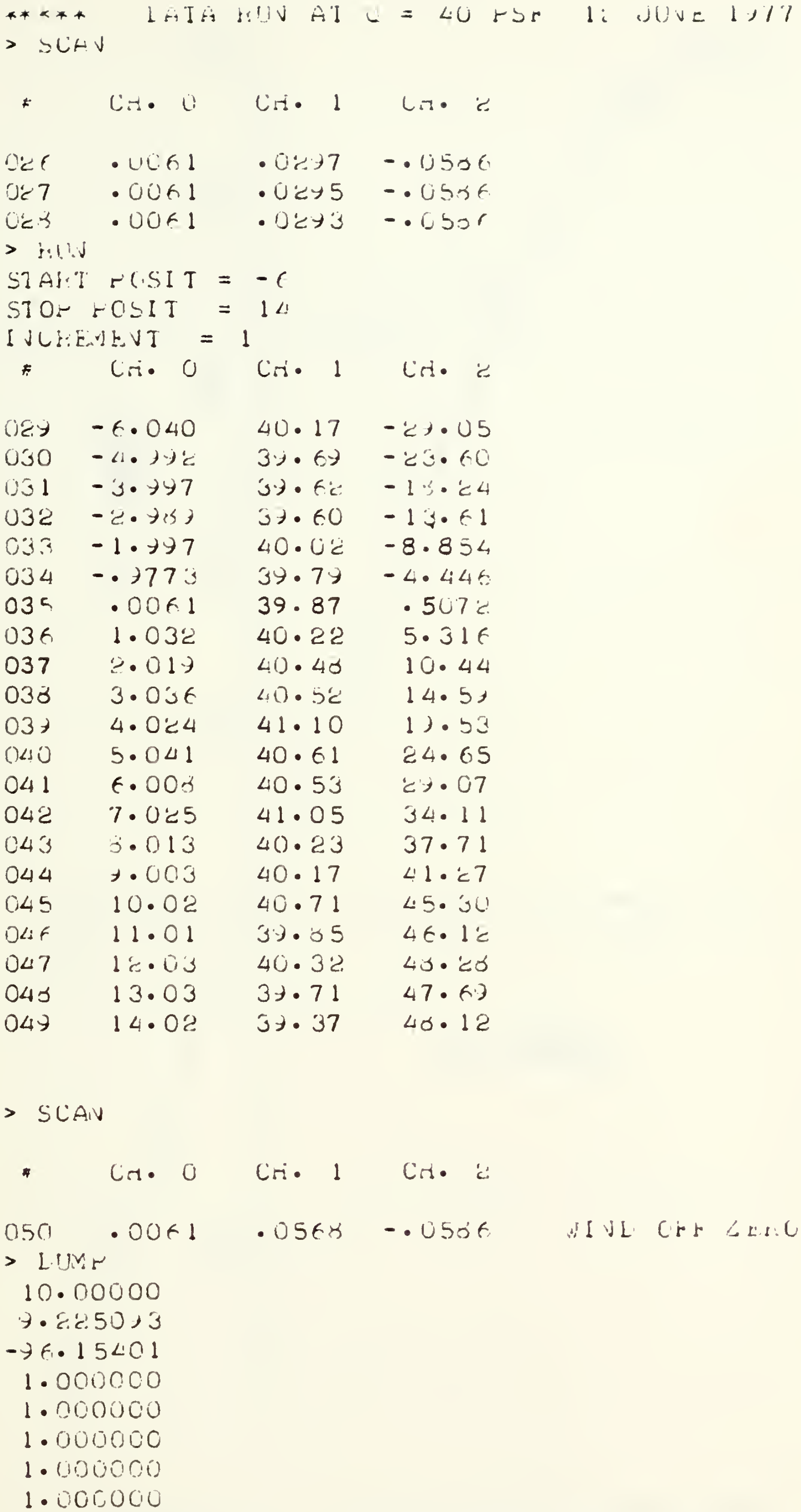
```
> EDIT
THIS COMMAND IS USED TO INPUT
TEXT WHICH IS USED MANY TIMES
DURING A DATA LOGGING SESSION.

THE "FILE" COMMAND IS USED TO
RECALL THE TEXT.


> FILE


THIS COMMAND IS USED TO INPUT
TEXT WHICH IS USED MANY TIMES
DURING A DATA LOGGING SESSION.

THE "FILE" COMMAND IS USED TO
RECALL THE TEXT.

> FILE


THIS COMMAND IS USED TO INPUT
TEXT WHICH IS USED MANY TIMES
DURING A DATA LOGGING SESSION.

THE "FILE" COMMAND IS USED TO
RECALL THE TEXT.

> EDIT
THE RUBOITTIUT IS USDGTYYTGDEFUL


> FILE


THE RUBOUT IS USEFUL

> @
*** RESET: ALL CHANNEL I/O IN "VOLTS" ***
> EDIT
TTTTTTTTTTTTTTTTTTTTTTTTTTTTTT

> EDIT
TTTTTT CNTRL-Z USED HERE
> FILE


TTTT CNTRL-Z USED HERETTTTTTTT
02: INVALID "FILE" TERMINATION
```

Figure 8 - EDIT AND FILE EXAMPLES



```
 > UNIT
  CHANNEL = 0
 UNIT/VOLT = 10
  CHANNEL = 1
 UNIT/VOLT = 999.225093
  CHANNEL = 2
 UNIT/VOLT = -96.154
  CHANNEL =
 > SET SCAN
 DELAY BETWEEN DATA POINTS = 15 MS (DEFAULT)
 "E" FORMAT(Y OR N) ? N
 INPUT CHANNELS IN DESIRED ORDER
 012

 WHEN READY TO TAKE DATA, TYPE  SCAN  OR RUN
>
*****    WIND OFF TARES
> SCAN

 #     CH. 0    CH. 1    CH. 2

001    .0061    .0115   -.2714       WIND OFF ZERO
> RUN
START POSIT = -6
STOP POSIT  = 14
INCREMENT   = 1
 #     CH. 0    CH. 1    CH. 2

002   -6.015    .0135   -.7822
003   -4.972    .0123   -.2274
004   -3.986    .0138   -.3127
005   -2.996    .0109   -.2283
006   -2.000    .0151   -.2036
007   -.9803    .0154   -.2384
008    .0061    .0112   -.3301
009    1.031    .0153   -.2421
010    2.016    .0160   -.2613
011    3.035    .0184   -.3035
012    4.022    .0212   -.2870
013    5.012    .0190   -.3017
014    6.017    .0184   -.2934
015    7.024    .0193   -.3145
016    8.013    .0221   -.1953
017    9.000    .0237   -.2366
018    10.02    .0239   -.2549
019    11.01    .0237   -.1678
020    12.03    .0285   -.1586
021    13.03    .0210   -.0953
022    14.02    .0217   -.0586
```

Figure 9 - WIND-TUNNEL TARE DATA



```
*****    DATA RUN AT Q = 40 PSF   12 JUNE 1977
> SCAN

 #      CH. 0    CH. 1    CH. 2

026    .0061    .0297   -.0586
027    .0061    .0295   -.0586
028    .0061    .0293   -.0586
> RUN
START POSIT = -6
STOP POSIT  = 14
INCREMENT  = 1
 #      CH. 0    CH. 1    CH. 2

029   -6.040    40.17   -29.05
030   -4.992    39.69   -23.60
031   -3.997    39.62   -18.24
032   -2.989    39.60   -13.61
033   -1.997    40.02   -8.854
034   -.9773    39.79   -4.446
035    .0061    39.87    .5072
036    1.032    40.22    5.316
037    2.019    40.48    10.44
038    3.036    40.52    14.59
039    4.024    41.10    19.53
040    5.041    40.61    24.65
041    6.008    40.53    29.07
042    7.025    41.05    34.11
043    8.013    40.23    37.71
044    9.003    40.17    41.27
045    10.02    40.71    45.30
046    11.01    39.85    46.12
047    12.03    40.32    48.28
048    13.03    39.71    47.69
049    14.02    39.37    48.12


> SCAN

 #      CH. 0    CH. 1    CH. 2

050    .0061    .0568   -.0586     WIND OFF ZERO
> DUMP
 10.00000
 9.225093
-96.15401
 1.000000
 1.000000
 1.000000
 1.000000
 1.000000
```

Figure 10 - WIND-TUNNEL DATA RUN

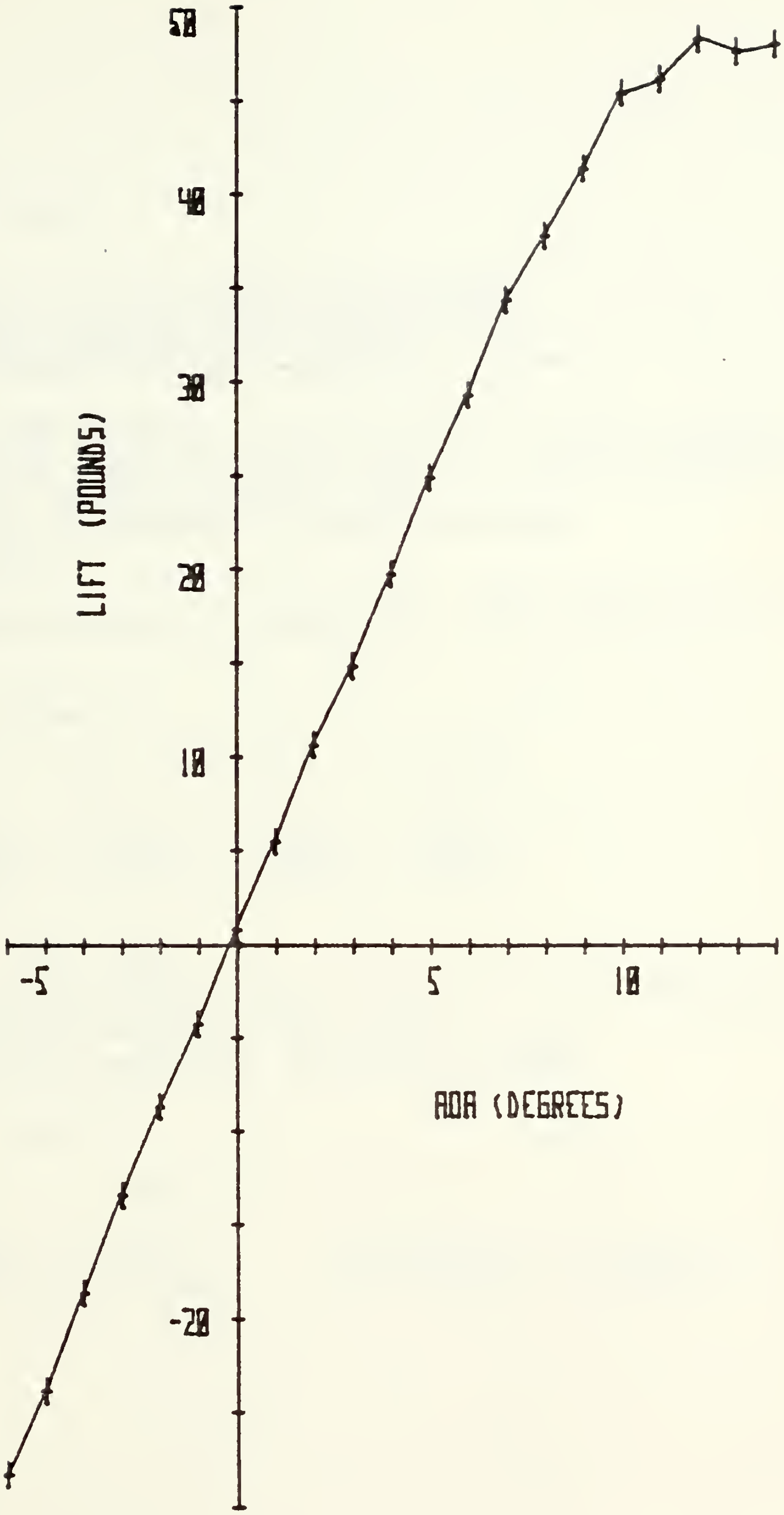
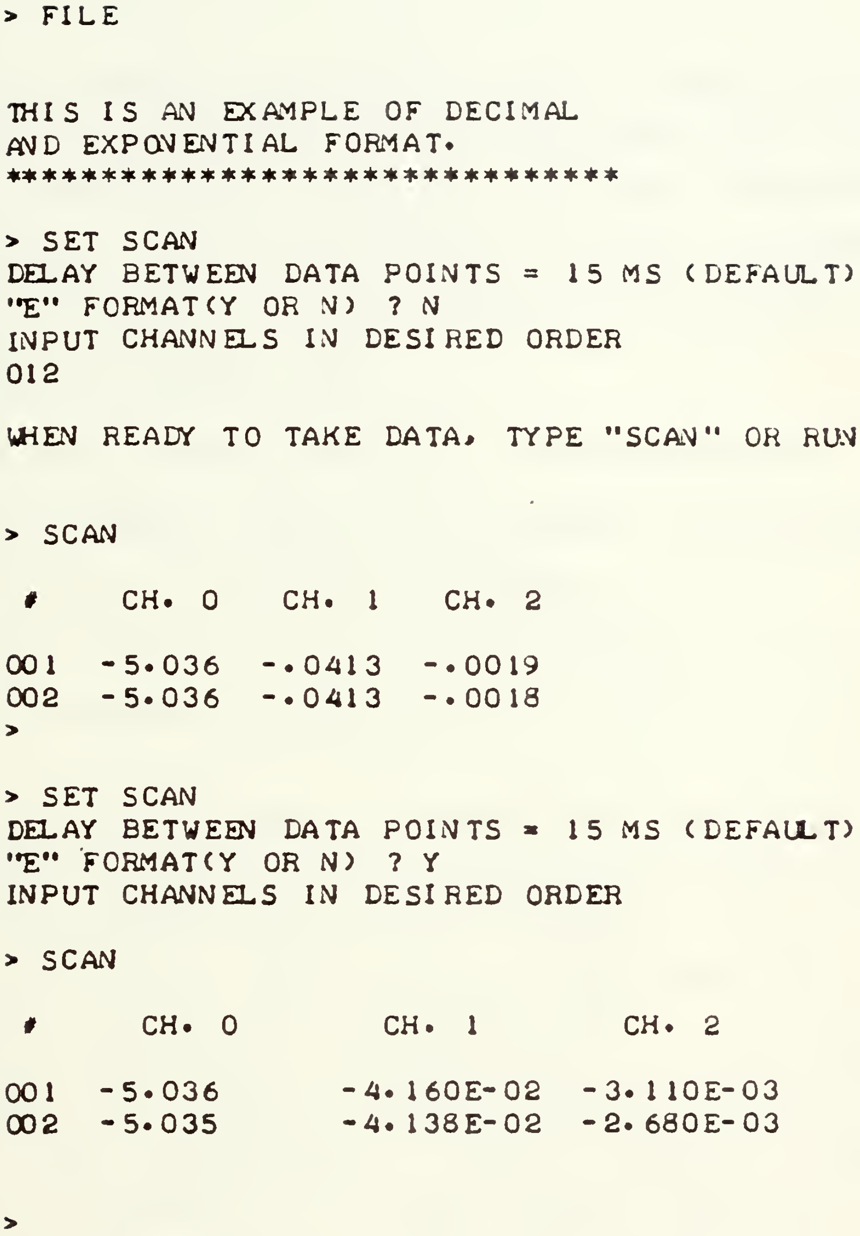


Figure 11 - PLOT OF WIND-TUNNEL DATA



```
> FILE


THIS IS AN EXAMPLE OF DECIMAL
AND EXPONENTIAL FORMAT.
******************************

> SET SCAN
DELAY BETWEEN DATA POINTS = 15 MS (DEFAULT)
"E" FORMAT(Y OR N) ? N
INPUT CHANNELS IN DESIRED ORDER
012

WHEN READY TO TAKE DATA, TYPE "SCAN" OR RUN


> SCAN

 #     CH. 0    CH. 1    CH. 2

001  -5.036   -.0413   -.0019
002  -5.036   -.0413   -.0018
>

> SET SCAN
DELAY BETWEEN DATA POINTS = 15 MS (DEFAULT)
"E" FORMAT(Y OR N) ? Y
INPUT CHANNELS IN DESIRED ORDER

> SCAN

 #      CH. 0        CH. 1        CH. 2

001  -5.036       -4.160E-02  -3.110E-03
002  -5.035       -4.138E-02  -2.680E-03


>
```

Figure 12 - EXPONENTIAL FORMAT EXAMPLES

IV. WIND TUNNEL APPLICATION

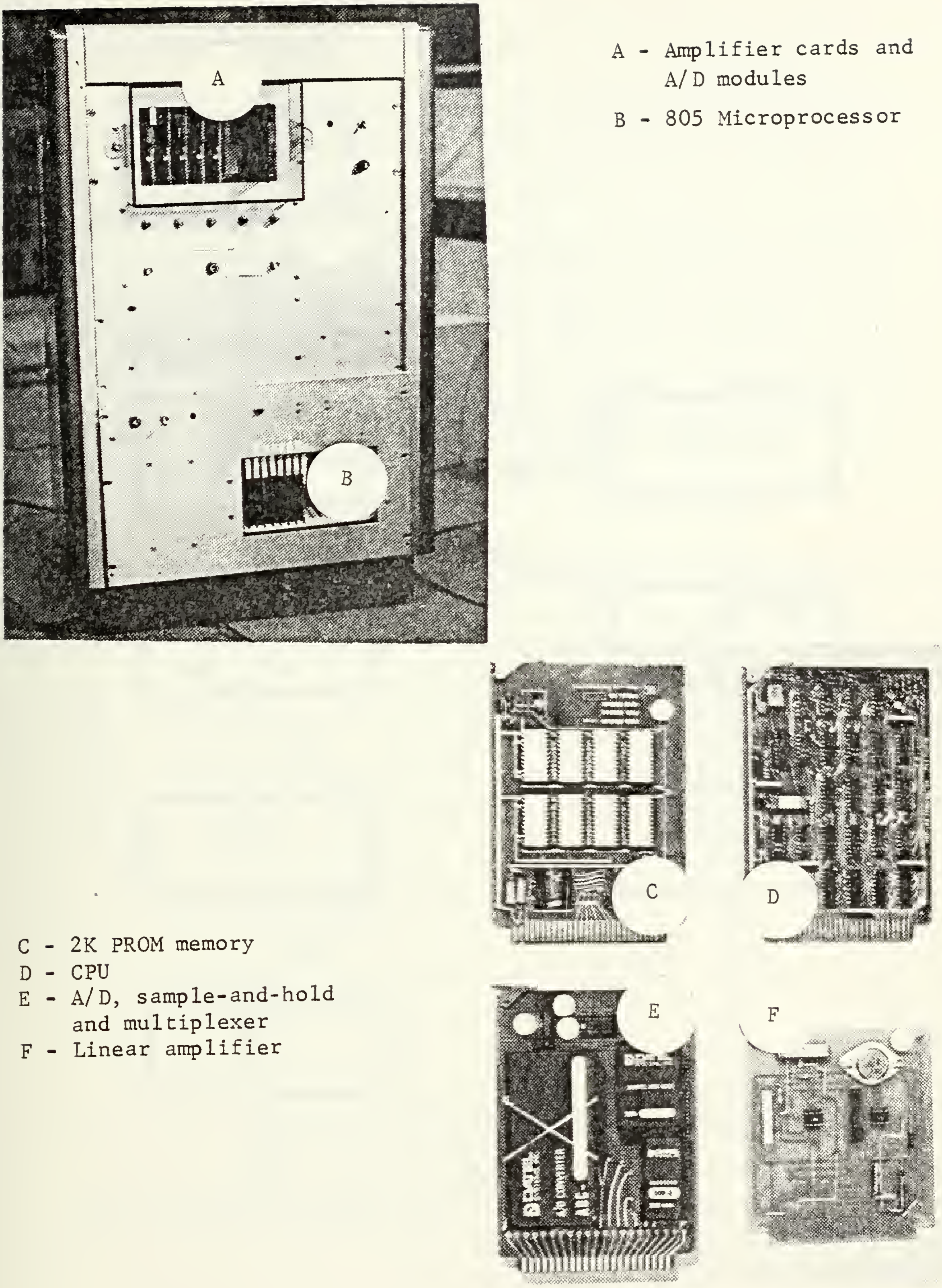
The ADL system was constructed in order to facilitate data acquisition and documentation from the 3.5 x 5.0 foot subsonic wind-tunnel located in the Department of Aeronautics at the Naval Postgraduate School. Logging data by hand frcm the tunnel balance is time consuming, error inducing and produces somewhat biased and scattered results. Other related problems are:

1. Personnel communications in a noisy environment.
2. Meter reading while the quantity to be measured is subjected to random perturbations.
3. Tunnel heating due to long run times.
4. Time consuming AOA setting.

The ADL effectively eliminated all the above problems and in addition it proved to be versatile enough to be used as a data logger with other equipment. Figure 13 is a picture of the ADL installation. It fits compactly into a roll-around cabinet and requires only a standard 20 mA current loop, 110 baud I/O device (e.g., a teletype), and patch cords to connect it to the voltage sources it is to monitor. Five variable-gain, linear amplifier cards are included to provide low level signal buffering. Voltage sources can be connected to the ADL directly or patched through an amplifier, as long as the input excursions do not exceed -10V to +10V.

The feedback control function is implemented via two output lines - one labeled UP and the other labeled DOWN.

Each line carries an independent logic level voltage which is used to actuate a relay. The two relays in turn are used to control the direction of a motor. The desired feedback quantity (in this case position) is input to channel 0 which closes the digital control loop. Figure 14 is a detailed schematic of the AOA control as used in the wind tunnel system. To date, the ADL was used to calibrate the wind tunnel balance and the dynamic pressure transducer [1].



A - Amplifier cards and
A/D modules

B - 805 Microprocessor

C - 2K PROM memory
D - CPU
E - A/D, sample-and-hold
and multiplexer
F - Linear amplifier

Figure 13 - PHOTOS OF ADL COMPONENTS

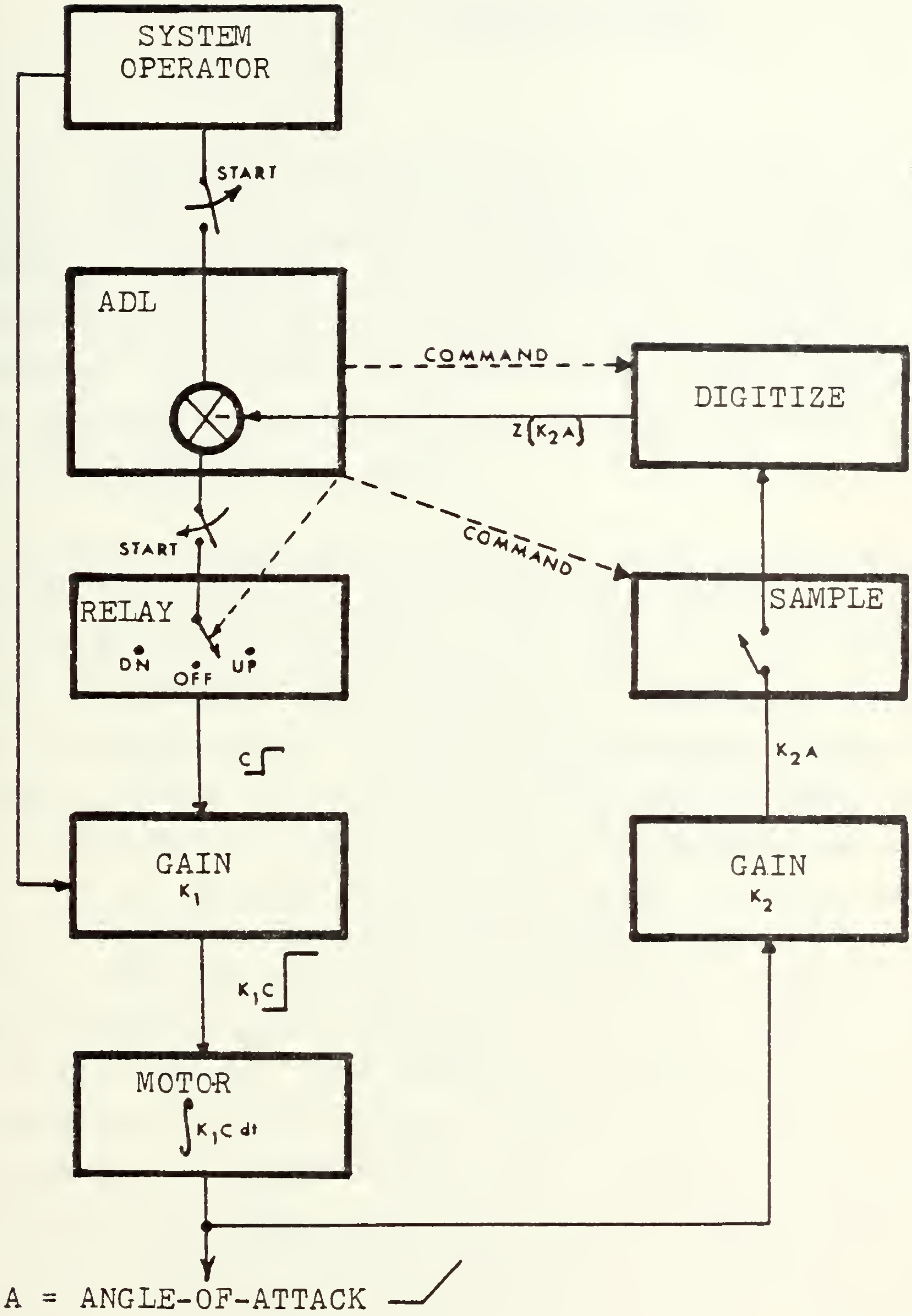


Figure 14 - ANGLE-OF-ATTACK FEEDBACK LOOP

V. SOFTWARE DESIGN

The assembled program listing for the ADL is presented in this chapter along with flowcharts for the most important routines. Throughout the following paragraphs, frequent reference is made to the 'position' of an external device. 'Position' is used for illustration; speed, angle or many other attributes of the state of an external device can be used as a feedback parameter.

Figure 15 shows the averaging process used in the filter routine. A running sum is taken at 128 data points. This sum is then divided by 128, converted back to binary and stored as a two byte quantity in registers D and E. The binary representation of the current desired position of the external device is recalled from memory and stored in registers B and C. Upon exit from the routine, the registers are set up to compare the actual and desired positions in order to determine in which direction to move the external device. The 'UP' driver is next shown. It makes use of the previous subroutine to determine when the desired position has been reached. The position correction routine in fig. 17 determines if the position arrived at by the UP or DOWN routines has met predefined error criteria. There are also routines for the DOWN direction that are identical to figs. 16 and 17 (except for the direction of movement). The MOVE routine is in fig 18; it provides the logic necessary to properly call the UP and DOWN routines. Figures 19 and 20 are flowcharts of the RUN routine. RUN provides the automatic control function of the ADL by calling SCAN and MOVE at external device positions defined by the user.

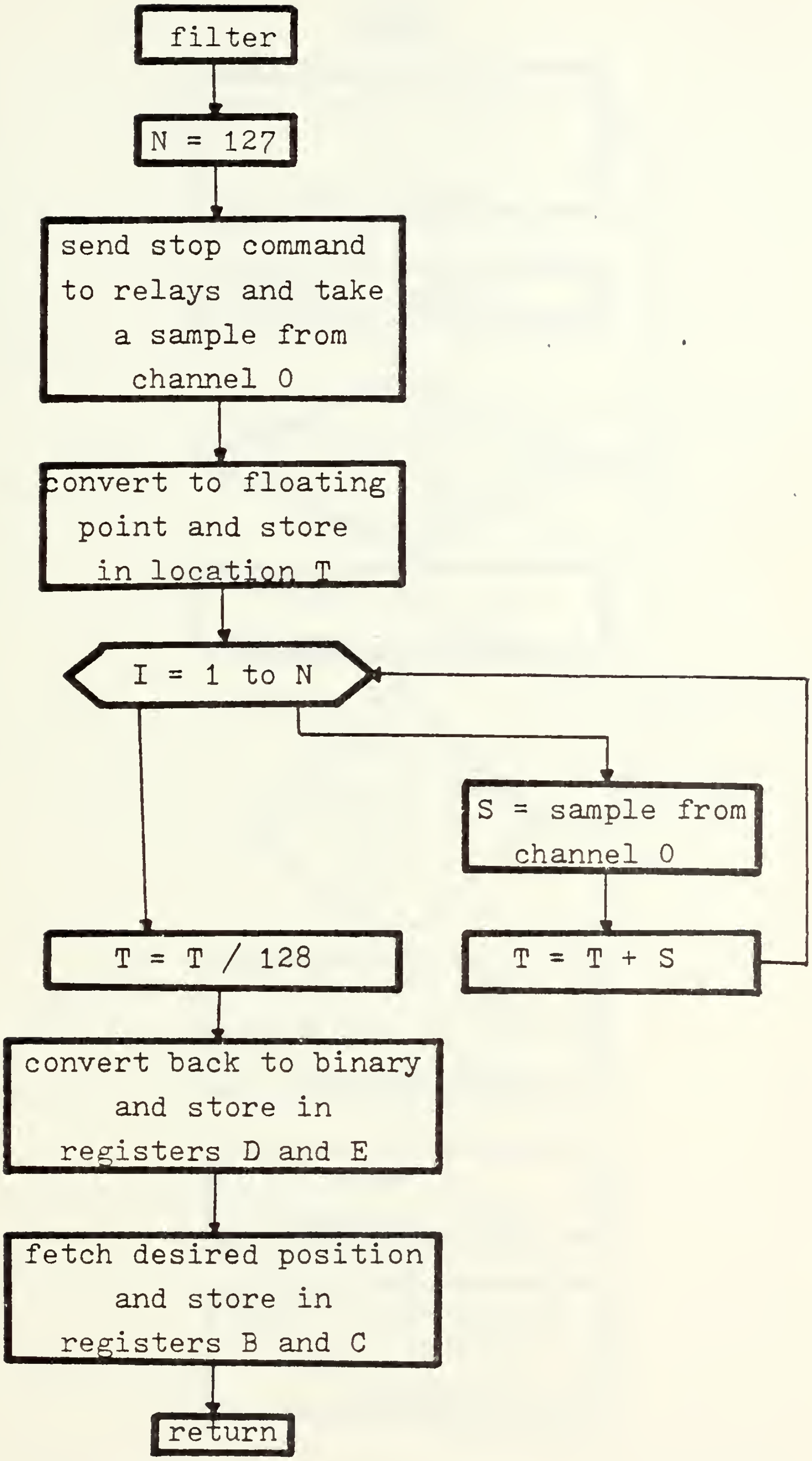

filter
N = 127
send stop command to relays and take a sample from channel 0
convert to floating point and store in location T
I = 1 to N
S = sample from channel 0
T = T + S
T = T / 128
convert back to binary and store in registers D and E
fetch desired position and store in registers B and C
return

Figure 15 - NOISE AND GLITCH FILTER LOGIC

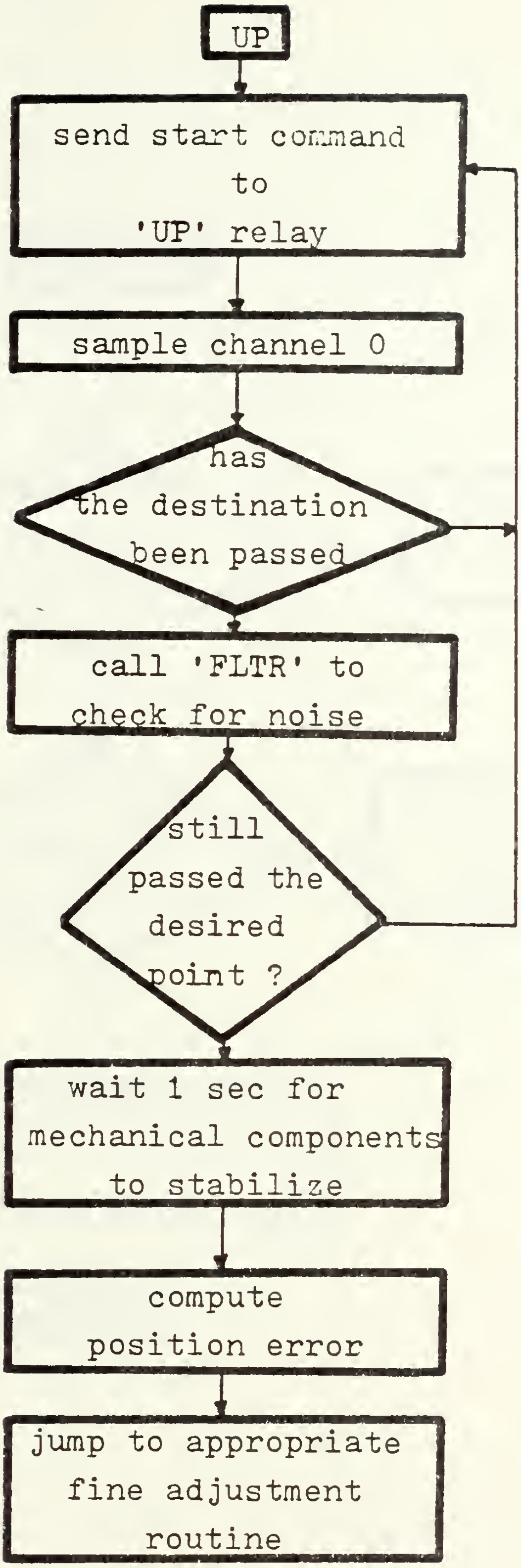

UP
send start command to 'UP' relay
sample channel 0
has the destination been passed
call 'FLTR' to check for noise
still passed the desired point ?
wait 1 sec for mechanical components to stabilize
compute position error
jump to appropriate fine adjustment routine

Figure 16 - 'UP' RELAY DRIVER LOGIC

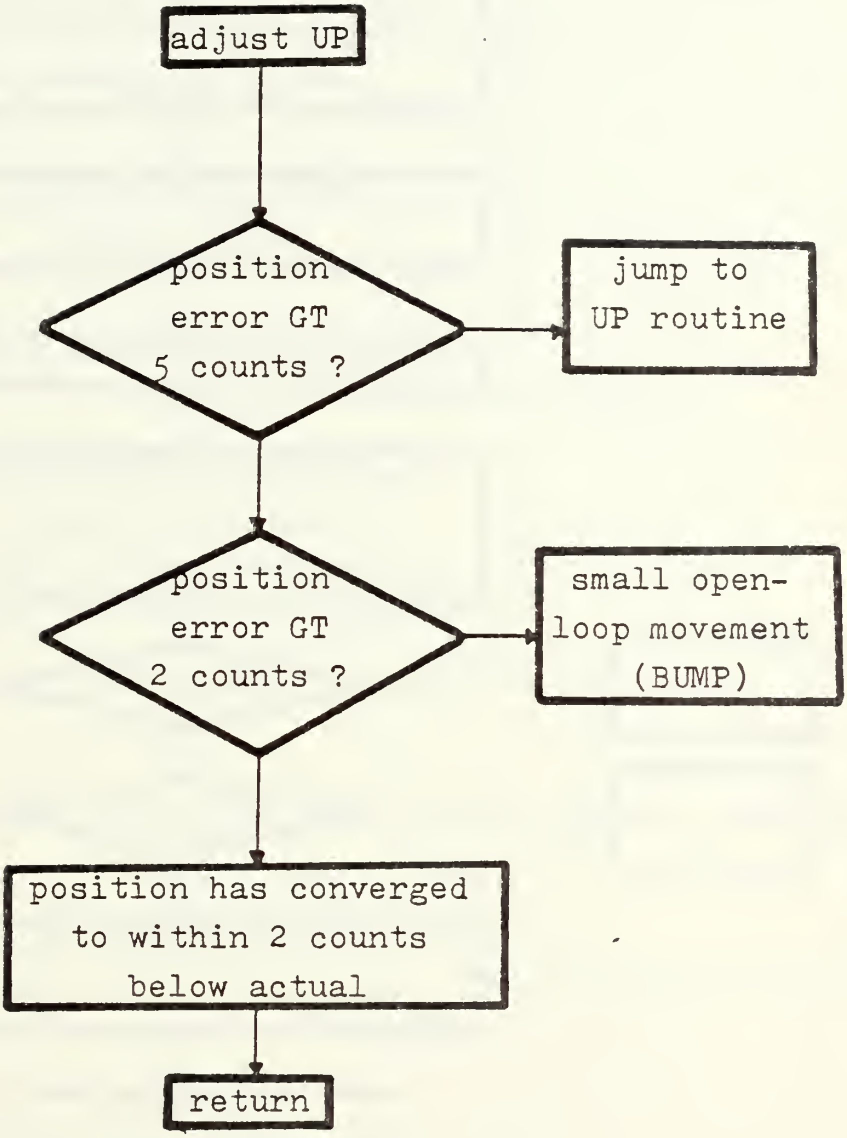

adjust UP
position error GT 5 counts ?
jump to UP routine
position error GT 2 counts ?
small open-loop movement (BUMP)
position has converged to within 2 counts below actual
return

Figure 17 - OVERSHOOT/UNDERSHOOT CORRECTION LOGIC

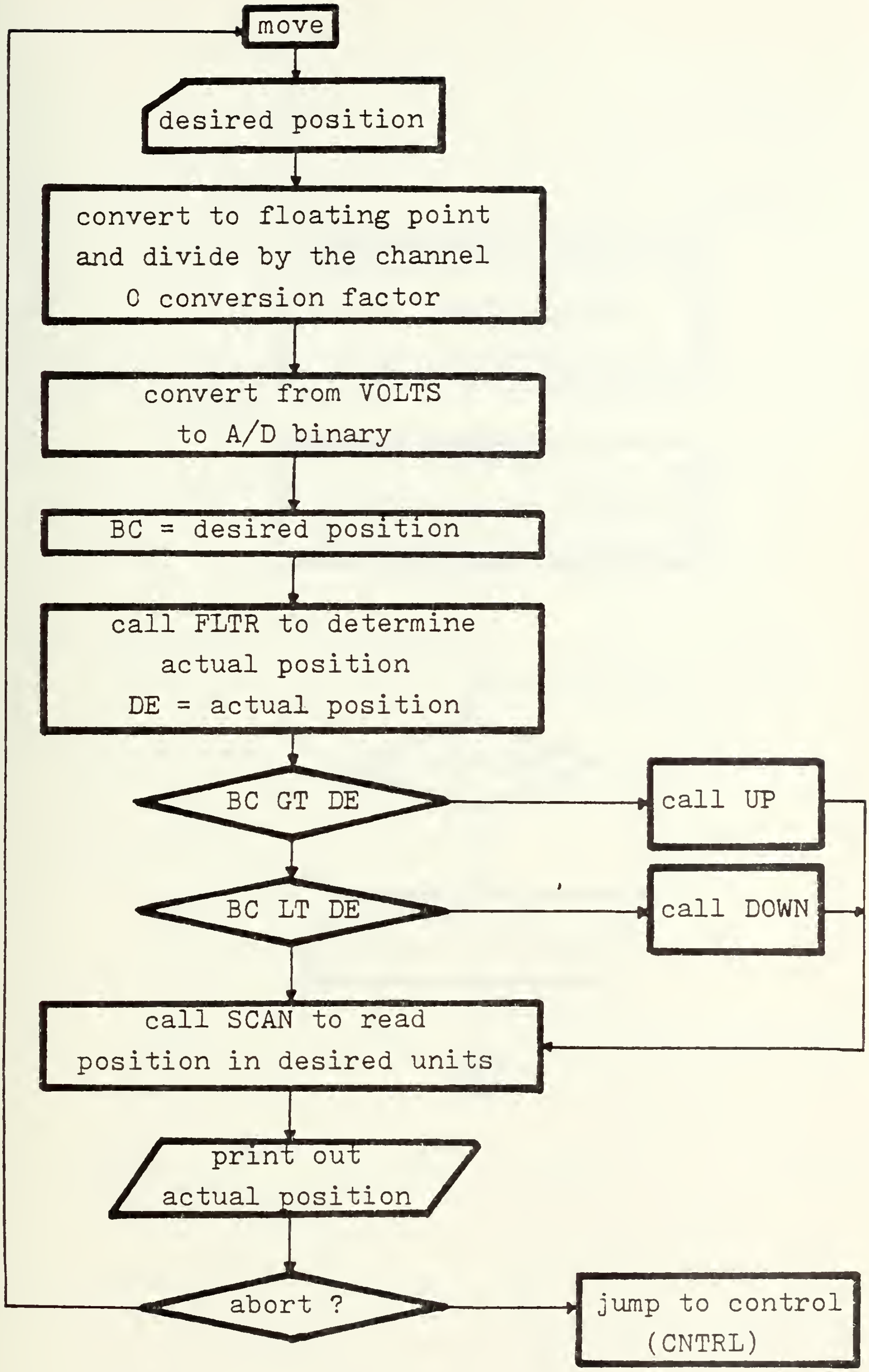

move
desired position
convert to floating point and divide by the channel 0 conversion factor
convert from VOLTS to A/D binary
BC = desired position
call FLTR to determine actual position DE = actual position
BC GT DE
call UP
BC LT DE
call DOWN
call SCAN to read position in desired units
print out actual position
abort ?
jump to control (CNTRL)

Figure 18 - EXTERNAL DEVICE CONTROL LOGIC

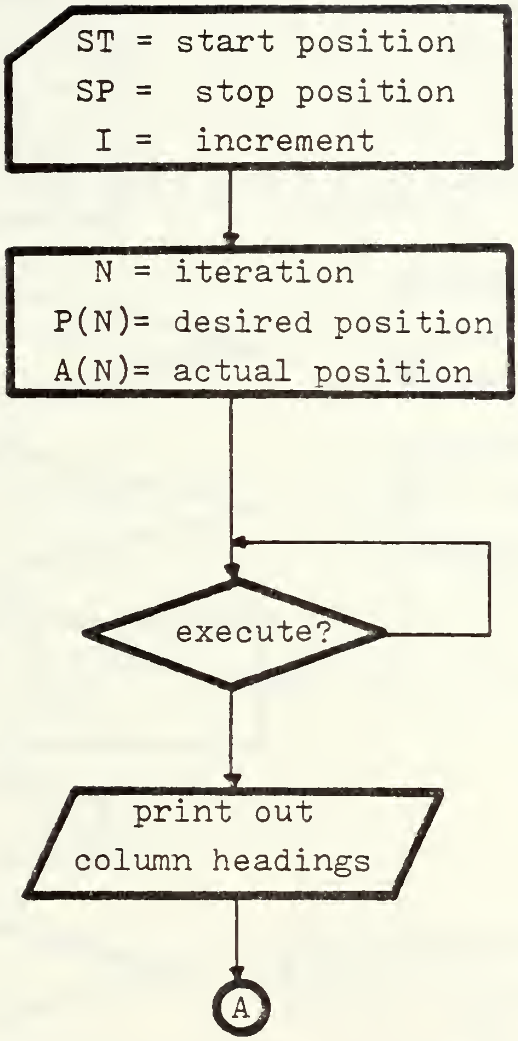


Figure 19 - RUN ROUTINE LOGIC (PART 1)

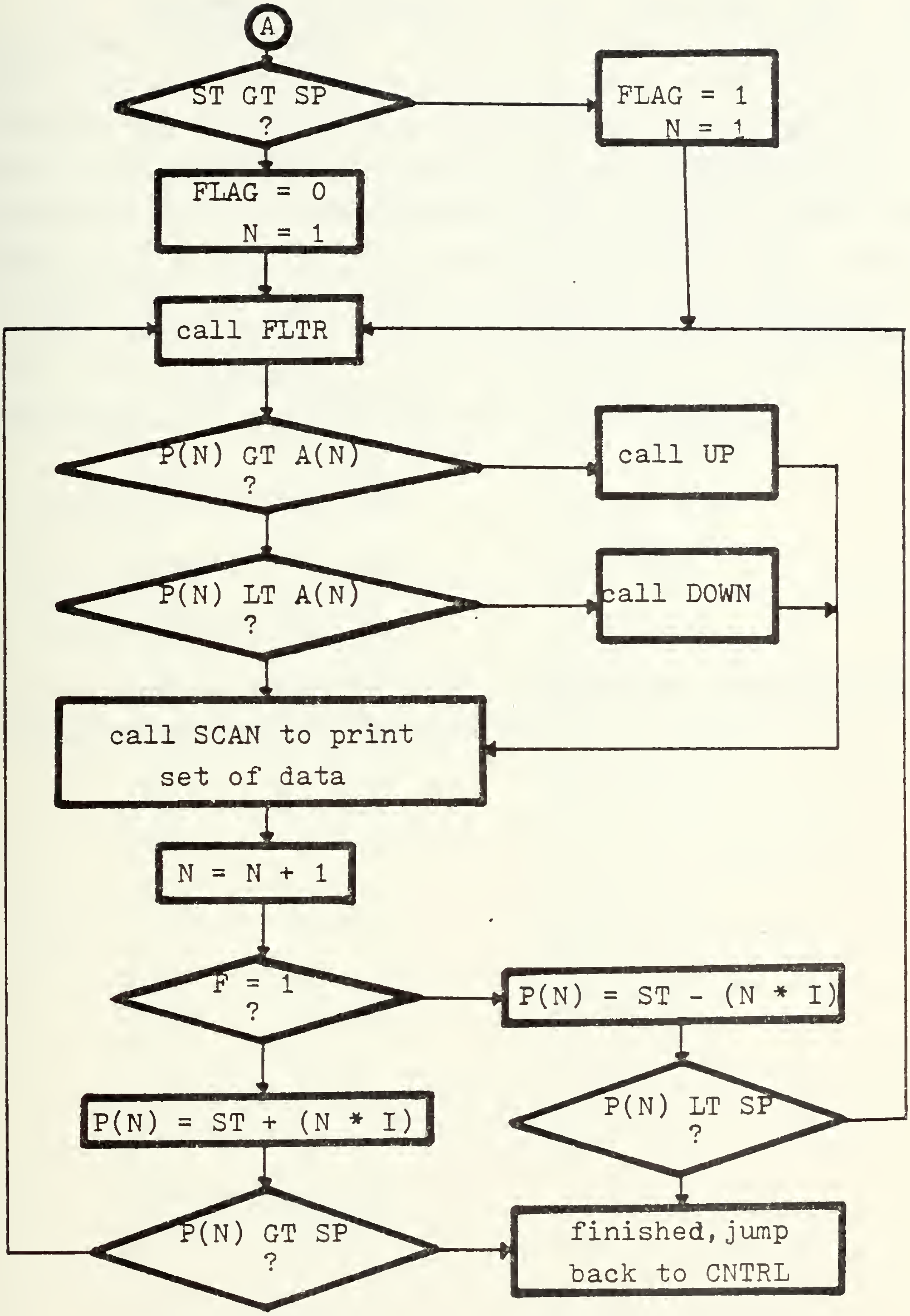


Figure 20 - RUN ROUTINE LOGIC (PART 2)

The following software was developed in small independent subroutines. This was done so that future revisions to logic could be accomplished with a minimum of redesign. For example, if a line printer is added to the system, all references to subroutine CO (console output) are changed so that the new routine can be called. The EQUATE tables at the start of each section of software are modified to reflect the address of the new routine, and the software is then reassembled. Similar procedures can be followed to alter I/O assignment, add new routines, etc. Memory requirements for the ADL software are as follows:

1. Pages 00H to 0FH, ROM

2. Pages 20H to 27H, ROM

3. Pages 10H to 1FH, RAM

This program was compiled on the INTELLEC 8 microprocessor development system for the 8008 U-P.

```
               ;
               ; UTILITY SUBROUTINES. THIS SECTION CONTAINS
               ; SUBROUTINES COMMON TO ALL MAJOR SECTIONS
               ; OF SOFTWARE.
               ;
0000                   ORG 0
               ;
032F           INIT    EQU 032FH        ;F.P. INITIALIZATION
036E           LOD     EQU 036EH        ;LOAD F.P. ACCUM
033E           STR     EQU 033EH        ;PLACE CONTENTS OF
                                        ;F.P. ACCUM INTO
                                        ;MEMORY
064B           INN     EQU 064BH        ;CHANGE BCD DATA TO
                                        ;F.P. AND LOAD ACCUM.
070F           OUU     EQU 070FH        ;CHANGE F.P. NUMBER
                                        ;TO BCD DATA STRING
03D7           AD      EQU 03D7H        ;F.P. ADDITION
03D4           SB      EQU 03D4H        ;F.P. SUBTRACTION
038C           MUL     EQU 038CH        ;F.P. MULTIPLY
03B4           DIV     EQU 03B4H        ;F.P. DIVIDE
0001           ABT     EQU 01H          ;COMMAND EXIT
                                        ;(CONTROL A)
000D           SCHAR   EQU 0DH          ;STOP CHAR
001A           SUBT    EQU 1AH          ;CHARACTER SUBSTITUTION
                                        ;(CONTROL Z)
10A0           STACK   EQU 10A0H        ;VARIABLE STORAGE
10FD           JUMP    EQU 10FDH        ;VECTORED JUMP
02B7           TEST    EQU 02B7H        ;LIGHT TEST
02CA           BLANK   EQU 02CAH        ;DISPL BLANKING
1040           DOPND   EQU 1040H        ;DECIMAL OPERAND BUFFER
1058           STORE   EQU 1058H        ;TEMP BUFFER
1070           RESLT   EQU 1070H        ;DECIMAL ANS BUFFER
0F00           TABLE   EQU 0F00H        ;TABLE OF VALID COMMAND
                                        ;CODES AND VECTORS.
                                        ;OPERATIONS
10C0           SCANB   EQU 10C0H        ;BUFFER CONTAINING
                                        ;CHANNEL SCAN INFO.
1080           CFBUF   EQU 1080H        ;CONVERSION FACTORS
007F           RBOUT   EQU 7FH          ;RUBOUT CHAR
0E8E           MTEST   EQU 0E8EH        ;RAM MEMORY TEST
               ;
               ; MESSAGES USED BY THIS SECTION ARE ANNOTATED
               ; AT THE END OF THE PROGRAM LISTING
               ;
2600           READY   EQU 2600H
2603           LERR    EQU 2603H
2648           LBOOT   EQU 2648H
               ;
0000 C0                DB 0C0H          ;FIRST INSTR = NOP
0001 06FF      QUIET:  MVI A,0FFH       ;RESET TTY BREAK
0003 51                OUT 8
0004 46B702            CALL TEST        ;CHECK DISPLAY
0007 464500    WAIT1:  CALL CI          ;WAIT FOR OPERATOR
000A 468E0E            CALL MTEST       ;CHECK RAM
```

```
000D 46CA02              CALL BLANK         ;CHECK DISPLAY
0010 462F03    BOOT:     CALL INIT          ;INITIALIZE MATH PAC
                                            ;AND DEFAULT VALUES.
0013 2E10367D            LXI H,RESLT+13
0017 3E0D                MVI M,SCHAR
0019 2E1036C0            LXI H,SCANB
001D 3E2A                MVI M,'*'          ;INVALID SCAN FLAG
001F 2E1036B0            LXI H,STACK+16     ;"WAIT" FLAG STORAGE
0023 3E2A                MVI M,'*'          ;15MS DEFAULT FLAG
0025 2E103680            LXI H,CFBUF        ;FILL CONVERSION FACTOR
                                            ;BUFFER WITH 1.0
0029 0E20                MVI B,32
002B 3E00      LOOP1:    MVI M,0
002D 30                  INR L
002E 09                  DCR B
002F 482B00              JNZ LOOP1
0032 2E103680            LXI H,CFBUF
0036 0E08                MVI B,8
0038 3E81      LOOP2:    MVI M,81H          ;1.0=81000000H
003A 0604                MVI A,4
003C 86                  ADD L
003D F0                  MOV L,A
003E 09                  DCR B
003F 483800              JNZ LOOP2
0042 448D00              JMP MON            ;START
               ;
               ; 'CI' CONSOLE INPUT ROUTINE
               ; INPUT: NO RESTRICTIONS
               ; REGISTERS: A,B,C,D
               ; OUTPUT: A,B ASCII
               ;            C,D = 0
0045 49        CI:       IN 4     ;GET START BIT
0046 1A                  RAR
0047 604500              JC CI
004A 466A00    START:    CALL HALF          ;1/2 DELAY
004D 1608                MVI C,8            ;BIT COUNT
004F 466500    RX:       CALL DELAY         ;CENTER OF NEXT BIT
0052 49                  IN 4
0053 1A                  RAR                ;ROT INTO CARRY
0054 C1                  MOV A,B            ;GET BUILD-UP WORD
0055 1A                  RAR
0056 C8                  MOV B,A            ;STORE
0057 11                  DCR C              ;C=C-1
0058 484F00              JNZ RX             ;CHECK FOR LAST BIT
005B 247F                ANI 7FH            ;MASK OFF PARITY
005D C8                  MOV B,A
005E 466500              CALL DELAY
0061 466500              CALL DELAY
0064 07                  RET
               ;
               ; 'DELAY' TTY DELAY LOOP
               ; DELAY PARAMETER IN 'D'
               ; PROVIDES 9MS DELAY.
0065 1EC5      DELAY:    MVI D,0C5H         ;1 BIT TIME
0067 446C00              JMP TIME
```

```
006A 1E62      HALF:   MVI D,062H       ;1/2 BIT TIME
006C 19        TIME:   DCR D
006D 486C00            JNZ TIME
0070 07                RET
               ;
               ; 'CRLF' OUTPUTS A CARRIAGE RET AND
               ; LINE FEED.
               ; INPUT: NO RESTRICTIONS
               ; REGISTERS: A,B,C,D
               ; OUTPUT: A=FFH,B=0AH;C,D=0
0071 0E0D      CRLF:   MVI B,0DH        ;CR
0073 467C00            CALL CO
0076 0E0A              MVI B,0AH        ;LF
0078 467C00            CALL CO
007B 07                RET
               ;
               ; 'CO' CONSOLE OUTPUT ROUTINE
               ; INPUT: WORD IS 7 BIT ASCII
               ;        STORED IN B
               ; REGISTERS: A,B,C,D
               ; OUTPUT: A=FFH, B IS SAVED, C,D = 0
               ;
007C C1        CO:     MOV A,B          ;FETCH WORD
007D B0                ORA A            ;CLEAR CARRY
007E 160B              MVI C,11         ;BIT COUNTER
0080 12                RAL              ;START BIT
0081 51        SEND:   OUT 8            ;TX TO TTY
0082 466500            CALL DELAY       ;WAIT 1 BIT TIME
0085 1A                RAR              ;POSITION NEXT BIT
0086 3CFF              CPI 0FFH         ;SET STOP BIT
0088 11                DCR C            ;C=C-1
0089 488100            JNZ SEND         ;SEND IF NOT DONE
008C 07                RET
               ;
               ; MONITOR ENTRY POINT AFTER POWER ON OR RESET.
               ;
008D 467100    MON:    CALL CRLF        ;RESET CARRIAGE
0090 2E263648          LXI H,LBOOT      ;START INFORMATION
0094 46E800            CALL LIST
0097 467100    CNTRL:  CALL CRLF
009A 2E263600          LXI H,READY      ;ACK
009E 46E800            CALL LIST
               ;
               ; GET COMMAND WORD AND FORM JUMP VECTOR
               ;
00A1 46F800    RECOG:  CALL GET
00A4 0E00              MVI B,0          ;INIT CHECK SUM
00A6 2E103640          LXI H,DOPND      ;POINT TO INPUT BUFF
00AA C7        RLOOP:  MOV A,M          ;FETCH CHARACTER
00AB 3C0D              CPI SCHAR        ;IF DONE SEARCH
00AD 68B600            JZ SRCH          ;  LOOK-UP TABLE
00B0 81                ADD B            ;ELSE BUILD CHECK SUM
00B1 C8                MOV B,A          ;STORE CK SUM
00B2 30                INR L
```

```
00B3 44AA00               JMP RLOOP
00B6 2E0F3600 SRCH:       LXI H,TABLE       ;POINT TO LOOK-UP
00BA C7       SRCHL:      MOV A,M
00BB 3C00                 CPI 0             ;VALIDITY CHECK
00BD 68DE00               JZ ERR
00C0 B9                   CMP B             ;COMPARE CHECK SUM WITH
00C1 68CA00               JZ VCTR           ;  TABLE. IF TRUE, JMP.
00C4 30                   INR L             ;ELSE GET NEXT
00C5 30                   INR L
00C6 30                   INR L
00C7 44BA00               JMP SRCHL
00CA 30       VCTR:       INR L             ;POINT TO LOW ADD
00CB CF                   MOV B,M           ;SAVE
00CC 30                   INR L             ;POINT TO HI ADD
00CD D7                   MOV C,M           ;SAVE
00CE 2E1036FD             LXI H,JUMP        ;POINT TO JUMP LCN
00D2 3E44                 MVI M, 44H
00D4 30                   INR L             ;LOAD VECTOR
00D5 F9                   MOV M,B
00D6 30                   INR L
00D7 FA                   MOV M,C
00D8 467100               CALL CRLF
00DB 44FD10   EXEC:       JMP JUMP          ;EXECUTE
00DE 2E263603 ERR:        LXI H,LERR        ;ERROR MSG OUT
00E2 46E800               CALL LIST
00E5 449700               JMP CNTRL         ;TRY AGAIN
              ;
              ; THIS ROUTINE IS USED TO OUTPUT
              ; STRINGS OF ALFA-NUM CHARACTERS
00E8 CF       LIST:       MOV B,M           ;OUT REGISTER
00E9 C7                   MOV A,M
00EA 3C0D                 CPI SCHAR         ;IF DONE RET
00EC 2B                   RZ
00ED 467C00               CALL CO           ;PRINT CHAR
00F0 30                   INR L             ;POINT TO NEXT
00F1 48E800               JNZ LIST          ;CHECK FOR PAGE WRAP
00F4 28                   INR H
00F5 44E800               JMP LIST          ;GET  NEXT
              ;
              ; 'GET' IS USED TO LOAD NUM DATA
              ; INTO DOPND OR LABELS INTO
              ; ANY DESIRED BUFFER. INSERTS
              ; 'CR' AS THE STOP CHAR.
              ; BUFFER CANNOT START AT XX00.
              ; DOES NOT ECHO A CARRIAGE RET.
              ; 'RUBOUT' ERASES PREVIOUSLY ENTERED
              ; CHARACTERS IN SUCCESSION.
              ; 'CONTROL Z' IS USED TO DISPLAY THE
              ; CONTENTS OF THE NEXT MEMORY LOCATION.
              ; RETURNS WITH LOW POINTER AT SCHAR.
              ;
00F8 2E103640 GET:        LXI H,DOPND       ;DEFAULT BUFFER
00FC E6       GETD:       MOV E,L           ;DESIRED BUFFER ENTRY
```

```
                                        ;SAVE LO POINTER
00FD 464500   CNTU:    CALL CI          ;TTY INPUT
0100 3C7F              CPI RBOUT        ;IF RUBOUT
0102 682C01            JZ ERASE         ;THEN ERASE
0105 3C01              CPI ABT          ;COMMAND EXIT
0107 689700            JZ CNTRL
010A 3C1A              CPI SUBT         ;DISPLAY NEXT CELL
                                        ;TO ALLOW FOR
                                        ;SUBSTITUTION.
010C 681E01            JZ CNTU4
010F 3C0D              CPI SCHAR        ;IF DONE RETURN
0111 481601            JNZ CNTU1
0114 F8                MOV M,A          ;STORE STOP CHAR
0115 07                RET
0116 467C00   CNTU1:   CALL CO          ;ECHO OUT
0119 F9                MOV M,B          ;LOAD INPUT
011A 30                INR L            ;POINT TO NEXT CELL
011B 44FD00            JMP CNTU         ;GET NEXT
011E C7       CNTU4:   MOV A,M          ;CHECK FOR EOL
011F 3C0D              CPI SCHAR
0121 6A7600            CZ CRLF+5        ;IF TRUE ,LINE-FEED
0124 CF                MOV B,M          ;DISPLAY CONTENTS OF
                                        ;CURRENT CELL
0125 467C00            CALL CO
0128 30                INR L            ;POINT TO NEXT CELL
0129 44FD00            JMP CNTU         ;GET NEXT
              ;
              ; 'ERASE' IS USED TO RUB OUT AN
              ; INCORRECTLY ENTERED CHAR.
012C 31       ERASE:   DCR L            ;POINT TO LAST INPUT
012D C6                MOV A,L          ;CHECK TO ENSURE
012E BC                CMP E            ;  INPUT BUFER WILL
012F 403301            JNC ECHO         ;  NOT UNDERFLOW
0132 30                INR L            ;  RESTORE LO POINT
0133 CF       ECHO:    MOV B,M          ;FETCH BAD CHAR
0134 467C00            CALL CO          ;REPEAT CHAR
0137 44FD00            JMP CNTU
              ;
              ; 'STRIP' CHANGES ASCII INTO
              ; SPECIAL BDC USED BY THE
              ; FLOATING POINT ROUTINE
              ; H,L MUST POINT TO BUFFER
              ;
013A C7       STRIP:   MOV A,M          ;FETCH
013B 3C0D              CPI SCHAR        ;IF DONE RET
013D 2B                RZ
013E 1430              SUI 30H          ;BCD CONVERSION
0140 F8                MOV M,A          ;STORE BCD
0141 30                INR L            ;POINT
0142 443A01            JMP STRIP        ;GET NEXT
              ;
              ; 'DISPY' CONVERTS SPECIAL BCD
              ; FROM THE FLOATING POINT ROUTINE
              ; TO ASCII.
```

```
0145 2E103670 DISPY:  LXI H,RESLT       ;POINT TO BUFF
0149 C7       DISPL:  MOV A,M           ;FETCH
014A 3COD             CPI SCHAR         ;IF DONE RET
014C 2B               RZ
014D 0430             ADI 30H           ;BCD TO ASCII
014F F8               MOV M,A           ;STORE
0150 30               INR L
0151 444901           JMP DISPL
              ;
              ; 'BINFP' CHANGES RAW BINARY DATA
              ; TO FLOATING POINT
0154 C7       BINFP:  MOV A,M           ;FETCH HI BYTE
0155 C8               MOV B,A
0156 30               INR L
0157 D7               MOV C,M           ;FETCH LO BYTE
0158 A0               ANA A             ;CLEAR CARRY
0159 2611             MVI E,17          ;RESET COUNT
015B 706D01   SHIFT:  JM EXIT           ;IF NEGATIVE
                                        ;DATA IS NORM
015E C8               MOV B,A           ;SAVE HI
015F 21               DCR E             ;E=E-1
0160 688201           JZ DZER           ;IS 0
0163 C2               MOV A,C           ;LOAD LO BYTE
0164 A0               ANA A             ;CLEAR CARRY
0165 12               RAL               ;TWO BYTE SHIFT
0166 D0               MOV C,A           ;TO THE LEFT
0167 C1               MOV A,B
0168 12               RAL
0169 A0               ANA A             ;SET CNTRL BITS
016A 445B01           JMP SHIFT         ;NEXT SHIFT
016D 247F     EXIT:   ANI 7FH           ;POS NUM MASK
016F 2E103659         LXI H,STORE+1
0173 F8               MOV M,A           ;MS FP BYTE
0174 30               INR L
0175 FA               MOV M,C           ;NEXT FP BYTE
0176 30               INR L
0177 3E00             MVI M,0           ;LS FP BYTE
0179 067F             MVI A,7FH         ;EXP ADJUST
017B 84               ADD E             ;BIAS-#SHIFTS
017C 2E103658         LXI H,STORE
0180 F8               MOV M,A           ;STORE EXP
0181 07               RET               ;NORMAL EXIT
0182 2E103658 DZER:   LXI H,STORE       ;DATA=0
0186 3E00             MVI M,0
0188 07               RET               ;0 EXIT
              ;
              ; 'FPBIN' CHANGES FLOATING POINT
              ; TO BINARY. HL MUST POINT TO HIGH BYTE
              ; OF FP DATA UPON ENTRY
              ; RAW RESULT IS IN DE.
0189 C7       FPBIN:  MOV A,M
018A 1480             SUI 80H           ;STRIP EXCESS 80H
018C C8               MOV B,A           ;SAVE
018D 0610             MVI A,16          ;2 BYTE BIAS
```

```
018F 91                 SUB B           ;COMPUTE # SHIFTS
0190 C8                 MOV B,A         ;SAVE
0191 30                 INR L
0192 C7                 MOV A,M
0193 3480               ORI 80H         ;FORM MSBYTE
0195 D8                 MOV D,A         ;SAVE IT
0196 30                 INR L
0197 E7                 MOV E,M         ;GET LSBYTE
0198 C3         CNTU3:  MOV A,D         ;RESTORE
0199 B0                 ORA A           ;SHIFT 2 BYTES
019A 1A                 RAR
019B D8                 MOV D,A
019C C4                 MOV A,E
019D 1A                 RAR
019E E0                 MOV E,A
019F 09                 DCR B           ;CHECK COUNTER
01A0 489801             JNZ CNTU3
01A3 07                 RET
0000             END
```

```
                ;
                ; DISPLAY LIGHT ROUTINES
                ; USED TO OUTPUT VOLTAGE DATA TO THE
                ; DISPLAY LITES. DATA IS OUTPUT IN
                ; 4 SIGNIFICANT FIGURES. THIS SECTION
                ; ALSO CONTAINS THE LITE TEST AND LITE
                ; BLANK FUNCTIONS WHICH ARE CALLED AS
                ; PART OF SYSTEM BOOT.
                ;
0000                    ORG 01C0H
                ;
                ; EQUATES NOT ANNOTATED IN THIS SECTION
                ; CAN BE FOUND IN PREVIOUS SECTIONS
                ;
1070            RESLT   EQU 1070H       ;DECIMAL RESULT
00FD            MINUS   EQU 0FDH        ;'-' - 30H
00FE            DECPT   EQU 0FEH        ;'.' - 30H
00F0            SPACE   EQU 0F0H        ;' ' - 30H
106C            SSTAT   EQU 106CH       ;SIGN STATUS STORAGE
006C            STAT    EQU 06CH        ;SSTAT POINTER
0030            PLO     EQU 030H        ;POS SIGN/LATCH OFF
000A            LT      EQU 0AH         ;LITE TEST COMMAND
00F0            DOM     EQU 0F0H        ;DEC POINT OFF MASK
00E0            DP0L    EQU 0E0H        ;DP IN LITE #0
00D0            DP1L    EQU 0D0H        ;DP IN LITE #1
00B0            DP2L    EQU 0B0H        ;DP IN LITE #2
0070            DP3L    EQU 070H        ;DP IN LITE #3
                ;
                ; OUTPUT SIGN AND STORE AS A STATUS WORD
                ;
01C0 2E103670   DISPL:  LXI H,RESLT     ;POINT TO RESULT
01C4 C7                 MOV A,M         ;FETCH SIGN
01C5 366C               MVI L,STAT
01C7 3CFD               CPI MINUS       ;IF - JUMP
01C9 68D201             JZ SETM
01CC 0610       SETP:   MVI A,10H       ;+ AND DISABLE
01CE 55                 OUT 2+8
01CF 44D501             JMP CNTU
01D2 0630       SETM:   MVI A,30H       ;- AND DISABLE
01D4 55                 OUT 2+8
01D5 F8         CNTU:   MOV M,A         ;STORE STATUS
                ;
                ; FETCH DIGITS FROM RESULT BUFFER,
                ; AND OUTPUT TO DISPLAY LIGHTS
                ;
01D6 3671               MVI L,71H       ;GET FIRST DIGIT
01D8 C7                 MOV A,M
01D9 3CFE               CPI DECPT
01DB 685802             JZ ZERO         ;IF DP, ADD LEADING 0'S
01DE DF                 MOV D,M         ;SAVE FIRST DIGIT
01DF 30                 INR L
01E0 C7                 MOV A,M         ;FETCH NEXT
01E1 3CFE               CPI DECPT
01E3 682602             JZ LZERO        ;IF DP, JUMP
```

```
01E6 0E00              MVI B,0         ;ELSE OUTPUT FIRST DIGIT
01E8 460902            CALL NODP
01EB 3CFE              CPI DECPT       ;CHECK 2ND DIGIT
01ED 687F02            JZ DP1
01F0 0E01              MVI B,1
01F2 460902            CALL NODP
01F5 3CFE              CPI DECPT
01F7 688A02            JZ DP2
01FA 0E02              MVI B,2
01FC 460902            CALL NODP
01FF 3CFE              CPI DECPT
0201 689502            JZ DP3
0204 0E03              MVI B,3
0206 440902            JMP NODP        ;EXIT
                 ;
                 ; OUTPUT A DIGIT WITH NO DP.
                 ;
0209 06F0        NODP:  MVI A,DOM
020B B3                 ORA D          ;ATTACH DATA
020C 461302             CALL LITE      ;OUT
020F DF                 MOV D,M        ;SAVE DIGIT
0210 30                 INR L
0211 C7                 MOV A,M        ;FETCH NEXT
0212 07                 RET
                 ;
                 ; OUTPUT AND LATCH
                 ; REGISTER B CONTAINS MUX ADD OF LITE
                 ;
0213 E6          LITE:  MOV E,L        ;SAVE POINT
0214 2CFF               XRI 0FFH
0216 57                 OUT 8+3        ;DATA OUT
0217 2E10366C           LXI H,SSTAT    ;GET STATUS
021B C7                 MOV A,M
021C B1                 ORA B          ;ATTACH MUX INFO
021D 55                 OUT 8+2        ;MUX AND SIGN OUT
021E 2C10               XRI 10H
0220 55                 OUT 8+2        ;SET LATCH
0221 2C10               XRI 10H
0223 55                 OUT 8+2        ;LATCH OFF
0224 F4                 MOV L,E        ;RESTORE
0225 07                 RET
                 ;
                 ; OUTPUT LEADING ZEROS IF NEEDED
                 ; TO CHANGE FROM SCIENTIFIC NOTATION
                 ; TO FIXED POINT.
                 ;
0226 367C        LZERO: MVI L,07CH
0228 C7                 MOV A,M        ;GET EXP
0229 3CF0               CPI SPACE
022B 483302             JNZ LZER1      ;CONTINUE IF NOT ' '
022E 3672               MVI L,072H     ;RESTORE
0230 445C02             JMP DP0
0233 DF          LZER1: MOV D,M
0234 3671               MVI L,71H
```

```
0236 0E00                MVI B,0
0238 06E0                MVI A,DPOL        ;0 + 1 DP
023A 461302              CALL LITE
023D 08                  INR B
023E 06F0                MVI A,DOM         ;0 + NO DP
0240 461302              CALL LITE
0243 19                  DCR D
0244 19                  DCR D
0245 689D02              JZ OUTZ2
0248 08                  INR B
0249 06F0                MVI A,DOM
024B 461302              CALL LITE
024E 19                  DCR D
024F 687702              JZ OUTZ1
0252 08                  INR B
0253 06F0                MVI A,DOM
0255 441302              JMP LITE          ;EXIT
0258 3671       ZERO:    MVI L,71H
025A 1E00                MVI D,0           ;DATA
                ;
                ; INDIVIDUAL DIGIT OUTPUTS
                ;
025C 0E00       DPO:     MVI B,0
025E 06E0                MVI A,DPOL
0260 B3                  ORA D             ;ATTACH DATA
0261 461302              CALL LITE
0264 30         OUT3:    INR L
0265 C7                  MOV A,M
0266 34F0                ORI DOM           ;ATTACH NO DP
0268 0E01                MVI B,1
026A 461302              CALL LITE
026D 30         OUT2:    INR L
026E C7                  MOV A,M
026F 34F0                ORI DOM
0271 0E02                MVI B,2
0273 461302              CALL LITE
0276 30         OUT1:    INR L
0277 C7         OUTZ1:   MOV A,M
0278 34F0                ORI DOM
027A 0E03                MVI B,3
027C 441302              JMP LITE          ;EXIT
027F 0E01       DP1:     MVI B,1
0281 06D0                MVI A,DP1L
0283 B3                  ORA D             ;ATTACH DATA
0284 461302              CALL LITE
0287 446D02              JMP OUT2
028A 0E02       DP2:     MVI B,2
028C 06B0                MVI A,DP2L
028E B3                  ORA D
028F 461302              CALL LITE
0292 447602              JMP OUT1
0295 0E03       DP3:     MVI B,3
0297 0670                MVI A,DP3L
0299 B3                  ORA D
```

```
029A 441302                JMP LITE         ;EXIT
029D C7         OUTZ2:     MOV A,M
029E 34F0                  ORI DOM
02A0 0E02                  MVI B,2
02A2 461302                CALL LITE
02A5 30                    INR L
02A6 447602                JMP OUT1
                ;
                ; OUTPUT CHANNEL NUMBERS
                ;
02A9 0E07       CLITE:     MVI B,7          ;LITE ADDRESS
02AB 34F0                  ORI DOM          ;ATTACH NO D.P.
02AD 461302                CALL LITE        ;OUTPUT CHANNEL #
02B0 0E06                  MVI B,6
02B2 06F0                  MVI A,DOM        ;OUTPUT '0'
02B4 441302                JMP LITE
                ;
                ; LIGHT TEST AND BLANK
                ; CALLED DURING POWER-UP OR RESET
                ; ACCUM CONTAINS THE DISPLAY DATA.
                ; REGISTER B CONTAINS THE MUX LITE NUMBER
                ;
02B7 0E07       TEST:      MVI B,7
02B9 2E10366C              LXI H,SSTAT
02BD 3E10                  MVI M,10H
02BF 1E0A                  MVI D,LT
02C1 C3         TESTL:     MOV A,D
02C2 461302                CALL LITE
02C5 09                    DCR B
02C6 50C102                JP TESTL
02C9 07                    RET
02CA 0E07       BLANK:     MVI B,7
02CC 2E10366C              LXI H,SSTAT
02D0 3E30                  M,PLO
02D2 1EFF                  MVI D,0FFH       ;TURN OFF LITES
02D4 44C102                JMP TESTL
0000             END
```

```
;

;
;
;
;
;
;
;
;
;                         PROGRAM SPACE 0300H TO 07FFH
;                         IS USED FOR THE FLOATING POINT
;                         PACKAGE. APPENDIX C PRESENTS
;                         EXCERPTS FROM THE INTEL
;                         USERS LIBRARY.
;
 END
```

```
                ;
                ; A/D SAMPLE AND HOLD
                ; THIS SECTION CONTAINS THE CODE TO PROPERLY
                ; DRIVE THE 'DATEL' (SEE TEXT) DATA
                ; ACQUISION MODULES. ALSO INCLUDED IS THE
                ; "READ" ROUTINE WHICH GENERATES THE
                ; VOLTMETER OUTPUT ON THE DISPLAY LIGHTS.
                ;
0000                   ORG 0800H
                ;
                ; EQUATES NOT ANNOTATED CAN BE FOUND IN
                ; PREVIOUS SECTIONS.
                ;
0009            CMD      EQU 9             ;OUTPUT PORT 1
0007            LBYTE    EQU 7             ;INPUT PORT 1
0005            MBYTE    EQU 5             ;INPUT PORT 2
106D            RAW      EQU 106DH         ;RAW DATA INPUT BUFFER
106F            MUXCH    EQU 106FH         ;DEFAUT MUX CHANNEL STORE
000D            SCHAR    EQU 0DH           ;STOP CHAR
00E8            LIST     EQU 00E8H         ;OUTPUT 1 LINE
                                           ;OF TEXT
0071            CRLF     EQU 0071H         ;OUTPUT CARRAGE RET-
                                           ;LINE FEED
00F8            GET      EQU 00F8H         ;INPUT DATA FROM
                                           ;OPERATOR
013A            STRIP    EQU 013AH         ;BCD←ASCII
0154            BINFP    EQU 0154H         ;F.P.←BIN
01C0            DISPL    EQU 01C0H         ;OUTPUT DATA TO
                                           ;LIGHT DISPLAY
02A9            CLITE    EQU 02A9H         ;OUTPUT CHAN # IN
                                           ;'A' TO DISPAY LIGHTS
0FF4            I2       EQU 0FF4H         ; 10.0
0FF8            I3       EQU 0FF8H         ; 819.15
0FFC            I4       EQU 0FFCH         ; 8191.5
1040            DOPND    EQU 1040H
1058            STORE    EQU 1058H
1070            RESLT    EQU 1070H
070F            OUU      EQU 070FH
036E            LOD      EQU 036EH
033E            STR      EQU 033EH
03D7            AD       EQU 03D7H
03D4            SB       EQU 03D4H
038C            MUL      EQU 038CH
03B4            DIV      EQU 03B4H
                ;
                ; MESSAGES
2672            LREAD    EQU 2672H
                ;
                ;
                ; 'SHOLD' EXECUTES ONE SAMPLE AND HOLD
                ; CYCLE, AND INPUTS A DOUBLE BYTE
                ; OF RAW DATA.
                ; INPUT: ACUM CONTAINS MUX CHANNEL
                ;        H,L POINT TO LSBYTE STORAGE
```

```
                  ; REGISTERS: A,L
                  ; OUTPUT: 'A' DESTROYED,   L=L-1
                  ;
0800 3410         SHOLD:  ORI 10H          ;RESET A/D MODULE
0802 53                   OUT CMD
0803 3418                 ORI 18H          ;SAMPLE
0805 53                   OUT CMD
0806 2C10                 XRI 10H          ;HOLD
0808 53                   OUT CMD
0809 4B           READ1:  IN MBYTE
080A 2440                 ANI 40H          ;CHECK FOR END OF CONVERT
080C 480908               JNZ READ1
080F 4F                   IN LBYTE         ;INPUT RAW DATA
0810 2CFF                 XRI 0FFH
0812 F8                   MOV M,A
0813 31                   DCR L
0814 4B                   IN MBYTE
0815 243F                 ANI 3FH
0817 2C3F                 XRI 3FH
0819 F8                   MOV M,A
081A 07                   RET
                  ;
                  ; "RVI" CHANGES RAW FLOATING POINT DATA
                  ; TO VOLTAGE UNITS PROPORTIONAL
                  ; TO THE RAW DATA FROM THE A/D. "VRI"
                  ; PERFORMS THE INVERSE OPERATION.
                  ;
081B 2E103658 RVI:        LXI H,STORE
081F 466E03               CALL LOD         ;A←M
0822 2E0F36F8             LXI H,I3
0826 46B403               CALL DIV         ;A←A/819.15
0829 2E0F36F4             LXI H,I2
082D 46D403               CALL SB          ;A←A-10.0
0830 07                   RET
0831 2E103658 VRI:        LXI H,STORE
0835 466E03               CALL LOD         ;A←M
0838 2E0F36F8             LXI H,I3
083C 468C03               CALL MUL         ;A←A*819.15
083F 2E0F36FC             LXI H,I4
0843 46D703               CALL AD          ;A←A+8191.5
0846 07                   RET
                  ;
                  ; 'READ' IS USED FOR CALIBRATION OF
                  ; ANY DESIRED CHANNEL. OUTPUT IS ON THE
                  ; DIGITAL DISPLAY IN VOLTAGE UNITS.
                  ;
0847 2E263672 READ:       LXI H,LREAD      ;PROMPT
084B 46E800               CALL LIST
084E 46F800               CALL GET         ;INPUT CH. NO.
0851 467100               CALL CRLF
0854 2E103640             LXI H,DOPND
0858 463A01               CALL STRIP       ;BCD←ASCII
085B 2E103640             LXI H,DOPND
085F C7                   MOV A,M
```

```
0860 2E10366F         LXI H,MUXCH    ;STORE CH. NO.
0864 F8               MOV M,A
0865 46A902           CALL CLITE     ;DISPLAY CH. NO.
0868 2E10366F NEXT:   LXI H,MUXCH    ;CONTINUE
086C C7               MOV A,M        ;RESTORE CH. NO.
086D 2E10366E         LXI H,RAW+1    ;POINT TO LSBYTE STORE
0871 460008           CALL SHOLD     ;GET DATA
0874 465401           CALL BINFP     ;CONVERT TO F.P.
0877 461B08           CALL RVI       ;CHANGE TO VOLTS
087A 2E103670         LXI H,RESLT
087E 460F07           CALL OUU       ;BCD←F.P.
0881 46C001           CALL DISPL     ;DISPLAY
0884 49               IN 4           ;SCAN TTY FOR INT
0885 1A               RAR            ;SET CARRY FLAG
0886 606808           JC NEXT        ;IF NO INT, GET
                                     ;MORE DATA FROM
                                     ;SAME CHANNEL
0889 467100           CALL CRLF      ;ELSE PROMPT
088C 444708           JMP READ
0000           END
```

```
                   ;
                   ; DATA ACQUISITION ROUTINES
                   ; THESE ROUTINES ARE USED TO UPDATE NUMERICAL
                   ; LABELS AND TO PROVIDE EDITING CAPABILITY
                   ; FOR ANNOTATING ANY GIVEN RUN.
                   ; IN ADDITION, THE "WAIT" ROUTINES
                   ; PROVIDE FOR VARIABLE TIME DELAYS BETWEEN
                   ; GROUPS OF SAMPLED DATA POINTS
                   ;
                   ;
0000                        ORG 08A0H
                   ;
                   ; EQUATES NOT ANNOTATED CAN BE
                   ; FOUND IN PREVIOUS SECTIONS
                   ;
000A               LF       EQU 0AH          ;LINE FEED
000D               SCHAR    EQU 0DH          ;STOP CHAR
1F00               BHEAD    EQU 1F00H        ;BUFF FOR "FILE"
0097               CNTRL    EQU 0097H        ;SYSTEM MONITOR
0065               DELAY    EQU 0065H        ;KILL TIME LOOP
007C               CO       EQU 007CH        ;CONSOLE OUT
00A1               RECOG    EQU 00A1H        ;COMMAND RECOGNITION
00FC               GETD     EQU 00FCH        ;INTO ANY BUFFER
0071               CRLF     EQU 0071H
00E8               LIST     EQU 00E8H
00F8               GET      EQU 00F8H
013A               STRIP    EQU 013AH
1040               DOPND    EQU 1040H
1070               RESLT    EQU 1070H
10A0               STACK    EQU 10A0H
00A5               NEWNO    EQU 0A5H         ;LSB OF COORD NO.
                   ;
                   ; MESAGES
                   ;
2603               LERR     EQU 2603H
267E               LWAIT    EQU 267EH
2629               LERR2    EQU 2629H
2793               LRUN     EQU 2793H
279C               LCOM     EQU 279CH
                   ;
                   ; 'COORD' UPDATES THE COORDINATION
                   ; NUMBER OF A GIVEN RUN.
                   ;
08A0 2E1036A5      COORD:   LXI H,STACK+5    ;LOAD COORDINATION
                                             ; NUMBER INTO
                                             ; A,B,C,D
08A4 C7                     MOV A,M
08A5 31                     DCR L
08A6 D7                     MOV C,M
08A7 31                     DCR L
08A8 DF                     MOV D,M
08A9 0401                   ADI 1            ;INCREMENT AND
                                             ; DECIMAL ADJUST
08AB 3C3A                   CPI 3AH          ;>= ASCII 10 ?
```

```
08AD 40B408             JNC ADJ1          ;JUMP IF TRUE
08B0 C8                 MOV B,A           ;ELSE RESTORE
08B1 44CD08             JMP DONE1         ;EXIT
08B4 0E30      ADJ1:    MVI B,'0'         ;RESET UNITS DIGIT
08B6 C2                 MOV A,C           ;CHECK 10'S DIGIT
08B7 0401               ADI 1
08B9 3C3A               CPI 3AH
08BB 40C208             JNC ADJ2
08BE D0                 MOV C,A
08BF 44CD08             JMP DONE1
08C2 1630      ADJ2:    MVI C,'0'         ;RESET 10'S DIGIT
08C4 C3                 MOV A,D           ;CHECK 100'S DIGIT
08C5 0401               ADI 1
08C7 3C3A               CPI 3AH
08C9 40EB08             JNC ADJ3
08CC D8                 MOV D,A
08CD 36A5      DONE1:   MVI L,NEWNO       ;STORE NEW NUMBER
08CF F9                 MOV M,B
08D0 31                 DCR L
08D1 FA                 MOV M,C
08D2 31                 DCR L
08D3 FB                 MOV M,D
08D4 CF                 MOV B,M           ;OUTPUT NEW NUMBER
08D5 467C00             CALL CO
08D8 30                 INR L
08D9 CF                 MOV B,M
08DA 467C00             CALL CO
08DD 30                 INR L
08DE CF                 MOV B,M
08DF 467C00             CALL CO
08E2 0E20               MVI B,' '         ;OUTPUT 2 SPACES
08E4 467C00             CALL CO
08E7 467C00             CALL CO
08EA 07                 RET               ;BACK TO CALLER
08EB 1E30      ADJ3:    MVI D,'0'         ;OVERFLOW
08ED 44CD08             JMP DONE1
               ;
               ; "RUNNO" PRINTS OUT 'RUN NO. '
               ; IT IS ENTERED WITH 'CONTROL R', AND
               ; ENABLES THE USER TO INSERT DESIRED
               ; RUN NUMBER INFO.
               ;
08F0 2E273693 RUNNO:    LXI H,LRUN        ;'RUN NO. '
08F4 46E800             CALL LIST
08F7 2E113600           LXI H,1100H       ;BLANK PAGE
08FB 46FC00             CALL GETD
08FE 449700             JMP CNTRL
               ;
               ; "CMNT" IS USED TO FLAG AND ENTER
               ; A ONE LINE COMMENT.
               ;
0901 2E27369C CMNT:     LXI H,LCOM        ;'*****  '
0905 46E800             CALL LIST
0908 2E113600           LXI H,1100H       ;BLANK PAGE
```

```
090C 46FC00              CALL GETD          ;INPUT COMMENT
090F 449700              JMP CNTRL          ;WHEN DONE, JUMP
                ;
                ; "EDIT" FORMATS THE "FILE" INFORMATION FOR
                ; LATER PRINT OUT. USES LF AS THE LAST
                ; ENTRY TO TERMINATE THE RECORD.
                ; "CONTROL F" IS USED TO EXIT THE ROUTINE ONLY
                ; AFTER EDITING AN EXISTING FILE. "CONTROL Z"
                ; IS USED TO STEP FOREWARD THROUGH AN EXISTING
                ; RECORD IN ORDER TO SUBSTITUTE CHARACTERS.
                ; "RUBOUT" IS USED TO STEP BACKWARD THROUGH AN
                ; EXISTING RECORD IN ORDER TO SUBSTITUTE
                ; CHARACTERS ("RUBOUT" ALWAYS PRECEEDS THE
                ; NEW CHARACTER STRING).
                ;
0912 2E1F3601 EDIT:      LXI H,BHEAD+1      ;TOP OF BUFFER
0916 46FC00   ELOOP:     CALL GETD
0919 467100              CALL CRLF
091C 31                  DCR L              ;FETCH LAST ENTRY
091D C7                  MOV A,M
091E 3C0A                CPI LF             ;IF LF, THEN DONE
0920 689700              JZ CNTRL
0923 30                  INR L              ;ELSE CONTINUE ENTRIES
0924 30                  INR L
0925 441609              JMP ELOOP
                ;
                ; 'HDR' PRINTS OUT THE HEADER WHICH WAS
                ; ENTERED BY THE ABOVE ROUTINE.
                ; IT IS ALSO USED TO OUTPUT MULTI-LINE
                ; RECORDS WHICH END WITH A SEPARATE LF CR
                ; SEQUENCE. CONTAINS AN OVERRUN PROTECTION
                ; TO PRENVENT AN INFINITE OUTPUT LOOP
                ; IN THE EVENT THAT THE FIRST CALL TO "EDIT"
                ; WAS ENDED WITH "CONTROL F" RATHER THEN LF.
                ; COMMAND WORD FOR ENTRY = "FILE"
                ;
0928 467100   HDR:       CALL CRLF
092B 467100              CALL CRLF
092E 2E1F3601            LXI H,BHEAD+1
0932 C7       HLOOP:     MOV A,M
0933 3C0D                CPI SCHAR          ;CHECK FOR EOL
0935 684B09              JZ NEXT
0938 3C0A                CPI LF             ;CHECK FOR EOR
093A 685109              JZ DONE2
093D C8                  MOV B,A            ;PRINT CHARACTER
093E 467C00              CALL CO
0941 30       HL1:       INR L              ;GET ANOTHER
0942 C6                  MOV A,L            ;CHECK FOR OVERRUN
0943 3C00                CPI 0
0945 685509              JZ ERR2
0948 443209              JMP HLOOP
094B 467100   NEXT:      CALL CRLF          ;EOL
094E 444109              JMP HL1
0951 467100   DONE2:     CALL CRLF
```

```
0954 07                 RET
              ;
0955 2E263629 ERR2:     LXI H,LERR2     ;ERROR MSG OUT
0959 46E800             CALL LIST
095C 07                 RET
              ;
              ; "FILE" IS USED AS THE ENTRY POINT FOR THE
              ; OUTPUT OF THE MULTI-LINE RECORD ENTERED
              ; WITH "EDIT"."HLOOP" IS THE ENTRY POINT
              ; FOR MULTI-LINE RECORDS POINTED TO
              ; WITH HL. ALL SUCH RECORDS MUST END WITH
              ; A LF CR SEQUENCE.
              ; IN ADDITION, ALL SUCH RECORDS MUST NOT
              ; CROSS PAGE BOUNDARIES.
              ;
095D 462809   FILE:     CALL HDR
0960 449700             JMP CNTRL
              ;
              ; "WAIT" IS USED TO STORE A DELAY PARAMETER
              ; WHICH IS USED BY "SCAN" IN ORDER TO
              ; PROVIDE A DELAY BETWEEN DATA POINTS.
              ;
0963 2E26367E WAIT:     LXI H,LWAIT
0967 463209             CALL HLOOP      ;PROMPT
096A 2E1036B0           LXI H,STACK+16  ;RESET WAIT FLAG
                                        ;TO TURN OFF
                                        ;DEFAULT OPTION
096E 3E00               MVI M,0
0970 44A100             JMP RECOG       ;GET WAIT FACTOR
                                        ;FROM OPERATOR
0973 2E1036AE MS25:     LXI H,STACK+14  ;STORE 25MS DELAY
0977 3E53               MVI M,83        ;FINE STORAGE
0979 31                 DCR L
097A 3E02               MVI M,2         ;COARSE DELAY
097C 449700             JMP CNTRL
097F 2E1036AE MS15:     LXI H,STACK+14  ;STORE 15MS DELAY
0983 3E62               MVI M,98
0985 31       EXIT:     DCR L
0986 3E01               MVI M,1
0988 449700             JMP CNTRL
098B 2E1036AE MS3:      LXI H,STACK+14  ;STORE 3MS DELAY
098F 3E17               MVI M,23
0991 448509             JMP EXIT
              ;
              ; "VWAIT" VARIABLE WAIT SUBROUTINE
              ; CALLED BY THE SCAN ROUTINE TO PROVIDE
              ; PROVIDE A DELAY BETWEEN DATA POINTS.
              ;
0994 2E1036AD VWAIT:    LXI H,STACK+13  ;COARSE DELAY
0998 E7                 MOV E,M         ;COUNTER
0999 30                 INR L           ;FINE DELAY
099A DF       VLOOP:    MOV D,M         ;FINE DELAY COUNTER
099B 466700             CALL DELAY+2
099E 21                 DCR E
```

```
099F 489A09              JNZ VLOOP
09A2 07                  RET
                ;
                ; THESE ROUTINES ARE USED TO IMPROVE
                ; OUTPUT READABILITY BY ROUNDING THE
                ; RESULT BUFFER TO 4 SIG DIGITS.
                ; "NOEX4" ASSUMES DATA NO LARGER THAN
                ; + OR - 9999, AND IS USED MAINLY
                ; FOR VOLTAGE OUTPUT. EITHER "YESEX"
                ; OR "NOEX4" MUST BE CALLED AFTER "ROUND".
                ;
                ;
                ; 'ROUND' IS USED TO OUTPUT 4 SIGNIFICANT
                ; DIGITS FROM THE DISPLAY REGISTER.
                ; A,B,E,H,L DESTROYED, NO INPUT RESTRICTIONS.
                ; MATH IS IN ASCII
                ;
                ;
09A3 2602       ROUND:   MVI E,2          ;ENTRY COUNTER
09A5 2E1036AF            LXI H,STACK+15   ;RESET OVERFLOW BIT
09A9 3E00                MVI M,0
09AB 2E103679            LXI H,RESLT+9
09AF 31         CONT1:   DCR L            ;POINT TO LSD
09B0 C7                  MOV A,M          ;GET IT
09B1 3C2E                CPI '.'          ;IF DP, JUMP
09B3 68BF09              JZ CONT2
09B6 3E0D                MVI M,SCHAR      ;ELSE INSERT CR
09B8 21                  DCR E            ;NEXT
09B9 48AF09              JNZ CONT1        ;IF E=0, DONE
09BC 44C609              JMP SIG          ;COMPUTE DIGITS
09BF 31         CONT2:   DCR L
09C0 3E30                MVI M,30H              ;INSERT ZERO
09C2 21                  DCR E            ;NEXT
09C3 48BF09              JNZ CONT2        ;IF E=0, DONE
09C6 31         SIG:     DCR L            ;POINT AT DIGIT
                                          ; TO BE OPERATED ON
09C7 C7                  MOV A,M          ;GET TRIAL DIGIT
09C8 3C2E                CPI '.'          ;IF DP GET NEXT
09CA 48D909              JNZ CONT3        ;ELSE CONTINUE
09CD 31                  DCR L
09CE C7                  MOV A,M
09CF 3E30                MVI M,30H        ;INSERT 0
09D1 3C35                CPI 35H
09D3 600A0A              JC EXIT5
09D6 44E009              JMP CONT5
09D9 3C35       CONT3:   CPI 35H          ;IF < 5 DO NOT
                                          ; ROUND
09DB 60070A              JC EXIT1
09DE 3E0D                MVI M,SCHAR      ;ELSE INSERT CR
09E0 2603       CONT5:   MVI E,3          ;DIGIT COUNTER
09E2 31         CONT4:   DCR L            ;POINT NEXT
09E3 C7                  MOV A,M          ;GET NEXT
09E4 3C2E                CPI '.'          ;IF DP GET NEXT
09E6 68E209              JZ CONT4
```

```
09E9 0401                ADI 1            ;ELSE INCR DIGIT
09EB 3C3A                CPI 3AH          ;IF < 10 DONE
09ED 60090A              JC EXIT2
09F0 3E30                MVI M,30H        ;ELSE RIPPLE CARRY
09F2 21                  DCR E            ;NEXT DIGIT
09F3 48E209              JNZ CONT4
                ;
09F6 31         LAST:    DCR L            ;GET LAST DIGIT
09F7 C7                  MOV A,M
09F8 3C2E                CPI '.'          ;IF DP GET NEXT
09FA 68F609              JZ LAST
09FD 0401                ADI 1            ;INCR DIGIT
09FF 3C3A                CPI 3AH          ;IF OVERFLOW JUMP
0A01 68130A              JZ EXIT3
0A04 44090A              JMP EXIT2        ;ELSE PRINT
                ;
0A07 060D       EXIT1:   MVI A,SCHAR      ;NORMAL EXIT
0A09 F8         EXIT2:   MOV M,A
0A0A 2E103670   EXIT5:   LXI H,RESLT      ;SIGN OUT
0A0E CF                  MOV B,M
0A0F 467C00              CALL CO
0A12 07                  RET
0A13 3E30       EXIT3:   MVI M,30H        ;INSERT 0
0A15 2E103670            LXI H,RESLT      ;DISPLAY BUFFER
0A19 CF                  MOV B,M          ;SIGN OUT
0A1A 467C00              CALL CO
0A1D 0E31                MVI B,'1'        ;OVERFLOW DIGIT
0A1F 467C00              CALL CO
0A22 2E1036AF            LXI H,STACK+15   ;SET OVERFLOW FLAG
0A26 3E01                MVI M,1
0A28 07                  RET
                ;
                ; "NOEX4" IS USED TO CHANGE FROM EXPONENTIAL
                ; FORMAT TO DECIMAL FORMAT FOR NUMBERS
                ; < .1 . MUST NOT BE USED FOR RESULTS
                ; >+ OR - 9999. AS THE ROUTINE ASSUMES ONLY
                ; SMALL NUMBERS ARE IN "E" FORMAT.
                ; USED TO IMPROVE THE READABILITY OF
                ; VOLTAGE OUTPUT.
                ; SIGN IS ASSUMED ALREADY OUT.
                ;
0A29 2E1036AF   NOEX4:   LXI H,STACK+15   ;CHECK FOR OVERFLOW
0A2D C7                  MOV A,M
0A2E 3C01                CPI 1
0A30 6A7E0A              CZ EXIT6         ;OUTPUT CARRY
0A33 2E103679            LXI H,RESLT+9    ;CHECK FOR E FORMAT
0A37 C7                  MOV A,M
0A38 3C45                CPI 'E'
0A3A 48850A              JNZ EXIT7        ;IF NO, NORMAL EXIT
0A3D 2E10367C            LXI H,RESLT+12   ;GET # DECIMAL PLACES
0A41 C7                  MOV A,M
0A42 2E1036AF            LXI H,STACK+15   ;SCRATCH
0A46 1431                SUI '1'          ;# LEADING 0'S
0A48 F8                  MOV M,A          ;SAVE
```

```
0A49 E0                 MOV E,A         ;COUNTER
0A4A 3C04               CPI 4
0A4C 406F0A             JNC ZERO        ;IF >3 O'S, NUMBER=O
0A4F 466F0A             CALL ZERO       ;OUTPUT LEADING O'S
0A52 2E1036AF           LXI H,STACK+15
0A56 CF                 MOV B,M         ;RESTORE
0A57 0604               MVI A,4         ;# SIG DIGITS
0A59 91                 SUB B           ;DIGITS REMAINING
0A5A E0                 MOV E,A         ;COUNTER
0A5B 2E103671 DGOUT:    LXI H,RESLT+1   ;SKIP SIGN
0A5F CF       DLOOP:    MOV B,M
0A60 467C00             CALL CO         ;DIGIT OUT
0A63 30       SKIPD:    INR L
0A64 C7                 MOV A,M
0A65 3C2E               CPI '.'         ;SKIP DP
0A67 68630A             JZ SKIPD
0A6A 21                 DCR E
0A6B 485F0A             JNZ DLOOP
0A6E 07                 RET
0A6F 0E2E     ZERO:     MVI B,'.'       ;LEADING O'S OUT
0A71 467C00             CALL CO
0A74 0E30               MVI B,'O'
0A76 467C00   ZLOOP:    CALL CO
0A79 21                 DCR E
0A7A 48760A             JNZ ZLOOP
0A7D 07                 RET
0A7E 2E103675 EXIT6:    LXI H,RESLT+5   ;TRUNCATE LS O
0A82 3E0D               MVI M,SCHAR
0A84 07                 RET
0A85 2E103671 EXIT7:    LXI H,RESLT+1   ;NORMAL EXIT
0A89 44E800             JMP LIST        ;RET THROUGH "LIST"
              ;
              ; "YESEX" IS USED TO RETAIN "E" FORMAT IN
              ; ORDER TO DISPLAY VERY LARGE OR VERY SMALL
              ; RESULTS. SIGN ASSUMED OUT.
              ;
0A8C 2E1036AF YESEX:    LXI H,STACK+15  ;CHECK FOR CARRY
0A90 C7                 MOV A,M
0A91 3C01               CPI 1
0A93 6A7E0A             CZ EXIT6
0A96 2E103671           LXI H,RESLT+1   ;OUTPUT MANTISSA
0A9A 46E800             CALL LIST
0A9D 2E103679           LXI H,RESLT+9   ;OUTPUT EXP
0AA1 44E800             JMP LIST
0000           END
```

```
             ;
             ;
             ; SCAN ROUTINES
             ; PROVIDES THE LOGIC NESSARY TO TAKE
             ; 128 SETS OF DATA POINTS FOR UP EIGHT
             ; CHANNELS OF DATA (IN ANY ORDER) WITH USER
             ; DEFINED TIME DELAY. THE RESULT IS CONVERTED
             ; TO UNITS DEFINED BY THE USER.
             ;
             ; EQUATES NOT ANNOTATED CAN BE FOUND
             ; IN PREVIOUS SECTIONS.
             ;
1050         FOPND   EQU 1050H         ;FLOATING POINT
                                       ;OPERAND BUFFER
10A0         PAGE    EQU 10A0H         ;HIGH ADD FOR RAW
                                       ;DATA STORAGE
10C0         SCANB   EQU 10C0H         ;CHANNEL SEQUENCE BUFF
00C0         SCB     EQU 0C0H          ;START OF SCAN BUFF
10A0         STACK   EQU PAGE          ;START OF VARIABLE
                                       ;SCRATCH PAD
10A1         LINE    EQU PAGE+1        ;LOW ADD FOR RAW
                                       ;DATA STORAGE
10A2         CHNPT   EQU LINE+1        ;POINTS TO A LCN IN
                                       ;THE SCAN STORAGE BUFF
0080         CFB     EQU 080H          ;START OF CF BUFFER
0800         SHOLD   EQU 0800H         ;SAMPLE/HOLD/CONVERT
                                       ;COMMANDS FOR A/D
0994         VWAIT   EQU 0994H         ;VARIABLE TIME DELAY
08A0         COORD   EQU 08A0H         ;UPDATE COORDINATION #
09A3         ROUND   EQU 09A3H         ;ROUND OUTPUT BUFFER
                                       ;TO 4 SIG DIGITS
0A29         NOEX4   EQU 0A29H         ;CONVERT SMALL NOS.
                                       ;FROM "E" FORMAT
                                       ;TO "F" FORMAT
0A8C         YESEX   EQU 0A8CH         ;RETAIN "E" FORMAT
                                       ;WITH 4 SIG DIGITS
                                       ;FOR NUMBERS LT .1
                                       ;OR GT 9999999.
0145         DISPY   EQU 0145H         ;ASCII←BCD
0045         CI      EQU 0045H         ;CONSOLE INPUT
081B         RVI     EQU 081BH         ;RAW DATA TO VOLTS
                                       ;INTERPOLATION
0FE4         I5      EQU 0FE4H         ;.0078125
00E8         LIST    EQU 00E8H
0071         CRLF    EQU 0071H
00F8         GET     EQU 00F8H
00FC         GETD    EQU 00FCH
1040         DOPND   EQU 1040H
1058         STORE   EQU 1058H
007C         CO      EQU 007CH
0097         CNTRL   EQU 0097H
0154         BINFP   EQU 0154H
036E         LOD     EQU 036EH
033E         STR     EQU 033EH
03D7         AD      EQU 03D7H
```

```
038C              MUL      EQU 038CH
1070              RESLT    EQU 1070H
070F              OUU      EQU 070FH
013A              STRIP    EQU 013AH
000D              SCHAR    EQU ODH
                  ;
0000                       ORG OABOH
                  ;
                  ; MESSAGES
                  ;
26A7              LSCAN    EQU 26A7H
26C7              INFO     EQU 26C7H
2610              LERR1    EQU 2610H
26F3              C1       EQU 26F3H
26F7              C2       EQU 26F7H
26FF              C3       EQU 26FFH
2704              DWAIT    EQU 2704H
2730              EXP      EQU 2730H
                  ;
                  ; 'SCAN3' TAKES 128 SETS OF DATA
                  ; POINTS AT VARIABLE INTERVALS.
                  ;
                  ; MACRO DEFINITIONS
                  ;
                                               ;INCREMENT A MEMORY
                  INRM     MACRO POINT,N       ;LOCATION N TIMES
                           LXI H,POINT
                           MOV A,M
                           ADI N
                           MOV M,A
                           ENDM
                  ;
                  ; COMMON SUBROUTINES
                  ;
OABO 2E1036A2 INDF:        LXI H,CHNPT         ;INDIRECT FETCH AND STORE
                                               ;IN 'A'. CHNPT CONTAINS
                                               ;LOW ADD. DATA ASSUMED
                                               ;ON SAME PAGE
OAB4 F7                    MOV L,M
OAB5 C7                    MOV A,M
OAB6 07                    RET
OAB7 2E1036A0 INDP:        LXI H,PAGE          ;INDIRECT POINTER
                                               ;STORED IN FIRST 2
                                               ;STACK POSITS.
OABB DF                    MOV D,M             ;SAVE
OABC 30                    INR L
OABD F7                    MOV L,M             ;LOW POINT
OABE EB                    MOV H,D             ;HIGH POINT
OABF 07                    RET
OACO 2E1036A0 SCAN2:       LXI H,PAGE          ;INITIALIZE
OAC4 3E12                  MVI M,12H           ;DATA INPUT BUFFER
OAC6 30                    INR L
OAC7 3E00                  MVI M,O             ;FIRST STORAGE LCN
OAC9 30                    INR L
```

```
0ACA 3EC0                MVI M,SCB           ;START OF INFO BUFFER
0ACC 07                  RET
               SKIP:     INRM CHNPT,1        ;IGNORE DELIMITER
0ACD 2E1036A2            LXI H,CHNPT
0AD1 C7                  MOV A,M
0AD2 0401                ADI 00001H
0AD4 F8                  MOV M,A

0AD5 46B00A               CALL INDF           ;TEST NEXT CHAR
0AD8 3C08                CPI 8
0ADA 40CD0A              JNC SKIP
0ADD 07                  RET
0ADE 2E1036A0  SCAN4:    LXI H,PAGE          ;RE-INIT BETWEEN SCANS
0AE2 3E12                MVI M,12H
0AE4 30                  INR L
0AE5 30                  INR L
0AE6 3EC0                MVI M,SCB
0AE8 07                  RET
0AE9 46B70A    DATA:     CALL INDP           ;POINT TO RAW STORAGE
0AEC 465401              CALL BINFP          ;CONVERT TO F.P.
0AEF 2E103658            LXI H,STORE         ;LOAD AND POINT
                                             ;  TO OPERAND
0AF3 466E03              CALL LOD
0AF6 2E103650            LXI H,FOPND
0AFA 07                  RET
               ;
               ;
               ;
0AFB 46B00A    SCAN3:    CALL INDF           ;GET DESIRED CHANNEL
0AFE 3C0D                CPI SCHAR
0B00 68220B              JZ COUNT            ;IF ALL CHANNELS SCANNED
                                             ;  SET UP NEXT STORAGE
0B03 3C08                CPI 8               ;A>=8 ?
0B05 42CD0A              CNC SKIP            ;IF TRUE, JUMP
0B08 46B70A              CALL INDP
0B0B 30                  INR L
0B0C 460008              CALL SHOLD          ;INPUT RAW DATA
                         INRM PAGE,1         ;NEXT CHANNEL
0B0F 2E1036A0            LXI H,PAGE
0B13 C7                  MOV A,M
0B14 0401                ADI 00001H
0B16 F8                  MOV M,A

                         INRM CHNPT,1        ;NEXT VECTOR POINTER
0B17 2E1036A2            LXI H,CHNPT
0B1B C7                  MOV A,M
0B1C 0401                ADI 00001H
0B1E F8                  MOV M,A

0B1F 44FB0A              JMP SCAN3           ;GET NEXT DATA
               COUNT:    INRM LINE,2         ;NEXT STORAGE
0B22 2E1036A1            LXI H,LINE
0B26 C7                  MOV A,M
0B27 0402                ADI 00002H
0B29 F8                  MOV M,A
```

```
0B2A 2B                RZ               ;CHECK FOR END
0B2B 469409            CALL VWAIT       ;KILL TIME
0B2E 46DE0A   RESET:   CALL SCAN4       ;GET ANOTHER SET
0B31 44FB0A            JMP SCAN3
              ;
              ;
              ;
              ; 'SCAN5' TAKES SETS OF DATA FROM THE
              ; CHANNEL ASSINGMENT DEFINED BY THE
              ; 'SET SCAN' ROUTINE.
              ;
              ;
0B34 46D10B   SCAN5:   CALL COLMN       ;PRINT COLUMN HEADINGS
0B37 467100            CALL CRLF
0B3A 07                RET              ;BACK TO CONTROLLER
0B3B 467100   RSCAN:   CALL CRLF
0B3E 46A008            CALL COORD       ;UPDATE COORDINATION
                                        ;NUMBER
0B41 46C00A   CNTU7:   CALL SCAN2       ;INIT FOR NEXT SCAN
                                        ;ALSO ENTRY POINT FOR
                                        ;SCANNING WITHOUT
                                        ;COL HEADINGS
0B44 46FB0A            CALL SCAN3       ;TAKE SET OF DATA
0B47 46C00A   DONE:    CALL SCAN2
0B4A 46B00A   AVE:     CALL INDF        ;GET CHANNEL
                                        ; COMPUTE AVERAGE
0B4D 3C0D              CPI SCHAR
0B4F 2B                RZ               ;RETURN TO CALLER
0B50 3C08              CPI 8
0B52 42CD0A            CNC SKIP
0B55 2E1036A6          LXI H,STACK+6    ;STORAGE FOR CONVERSION
                                        ;FACTOR VECTOR.
0B59 D0                MOV C,A          ;SAVE 'A'
0B5A B0                ORA A            ;CLEAR CARRY
0B5B 12                RAL              ;MPY BY 4
0B5C 12                RAL
0B5D 0E80              MVI B,CFB        ;START OF CONVERSION
                                        ;FACTOR BUFFER
0B5F 81                ADD B            ;COMPUTE VECTOR
0B60 F8                MOV M,A          ;STORE VECTOR
0B61 C2                MOV A,C          ;RESTORE 'A'
0B62 46E90A            CALL DATA        ;CONVERT AND STORE
0B65 463E03            CALL STR
              AVEL:    INRM LINE,2      ;NEXT RAW DATA
0B68 2E1036A1          LXI H,LINE
0B6C C7                MOV A,M
0B6D 0402              ADI 00002H
0B6F F8                MOV M,A

0B70 68830B             JZ NEXTP         ;IF DONE, OUT RESULTS
                                        ;SETUP FOR NEXT SCAN
0B73 46E90A            CALL DATA        ;CONVERT AND STORE
                                        ;RAW DATA POINT
0B76 46D703            CALL AD          ;ADD TO PREVIOUS
```

```
0B79 2E103650           LXI H,FOPND       ;STORE PARTIAL SUM
0B7D 463E03             CALL STR
0B80 44680B             JMP AVEL
0B83 2E0F36E4 NEXTP:    LXI H,I5          ;COMPUTE AVERAGE
                                          ;  AND CONVERT TO
                                          ;  VOLTAGE UNITS
0B87 466E03             CALL LOD          ;A←.0078125
0B8A 2E103650           LXI H,FOPND
0B8E 468C03             CALL MUL          ;A←A*SUM
0B91 462208             CALL RVI+7        ;A←VOLT(A)
0B94 2E1036A6           LXI H,STACK+6     ;CONVERSION FACTOR
                                          ;VECTOR
0B98 F7                 MOV L,M
0B99 468C03             CALL MUL          ;CHANGE FROM VOLTS
                                          ;TO USER DEFINED UNITS
0B9C 2E103670           LXI H,RESLT
0BA0 460F07             CALL OUU          ;CONVERT TO DECIMAL
0BA3 464501             CALL DISPY        ;CONVERT TO ASCII
0BA6 46A309             CALL ROUND        ;4 SIG DIGITS
0BA9 2E1036B1           LXI H,STACK+17    ;FORMAT CHECK
0BAD C7                 MOV A,M
0BAE 3C59               CPI 'Y'
0BB0 68250C             JZ EXP1           ;IF TRUE,"E" FORMAT
0BB3 46290A             CALL NOEX4        ;ELSE "F" FORMAT
0BB6 0E20      CNTU8:   MVI B,' '         ;OUTPUT 2 SPACES
0BB8 467C00             CALL CO
0BBB 467C00             CALL CO
                        INRM PAGE,1       ;P=P+1
0BBE 2E1036A0           LXI H,PAGE
0BC2 C7                 MOV A,M
0BC3 0401               ADI 00001H
0BC5 F8                 MOV M,A

                         INRM CHNPT,1      ;C=C+1
0BC6 2E1036A2           LXI H,CHNPT
0BCA C7                 MOV A,M
0BCB 0401               ADI 00001H
0BCD F8                 MOV M,A

0BCE 444A0B             JMP AVE
               ;
               ;
0BD1 2E1036A2 COLMN:    LXI H,CHNPT       ;LOAD POINTER WITH
                                          ;START OF SCAN BUFF
0BD5 3EC0               MVI M,SCB
0BD7 2E1036B1           LXI H,STACK+17
0BDB C7                 MOV A,M           ;CHECK FOR
                                          ;"E" FORMAT
0BDC 3C59               CPI 'Y'           ;YES ? THEN CONTINUE
0BDE 48F00B             JNZ CNTUD         ;ELSE JUMP
0BE1 2E2636F3           LXI H,C1          ;' # '
0BE5 46E800             CALL LIST
0BE8 0E0D               MVI B,SCHAR       ;CARRIAGE RETURN
0BEA 467C00             CALL CO
```

```
0BED 44F70B              JMP CLOOP
0BF0 2E2636F3 CNTUD:     LXI H,C1          ;' # '
0BF4 46E800              CALL LIST
0BF7 46B00A   CLOOP:     CALL INDF         ;GET CHANNEL
0BFA 3C0D                CPI SCHAR
0BFC 2B                  RZ                ;RET WHEN DONE
0BFD 3C08                CPI 8             ;IGNORE DELIMITERS
0BFF 42CD0A              CNC SKIP
0C02 0430                ADI 30H           ;CONVERT TO ASCII
0C04 E0                  MOV E,A           ;SAVE CH. NO.
0C05 2E1036B1            LXI H,STACK+17    ;CHECK FORMAT
0C09 C7                  MOV A,M
0C0A 3C59                CPI 'Y'
0C0C 682B0C              JZ EXP2           ;IF TRUE, "E" FORMAT
                                           ;ADJUST HEADINGS
0C0F 2E2636F7 CNTU9:     LXI H,C2          ;'   CH. '
0C13 46E800              CALL LIST
0C16 CC                  MOV B,E           ;RECALL CH. NO.
0C17 467C00              CALL CO           ;OUTPUT
                         INRM CHNPT,1      ;C=C+1
0C1A 2E1036A2            LXI H,CHNPT
0C1E C7                  MOV A,M
0C1F 0401                ADI 00001H
0C21 F8                  MOV M,A

0C22 44F70B               JMP CLOOP
0C25 468C0A   EXP1:      CALL YESEX        ;"E" FORMAT
0C28 44B60B              JMP CNTU8
0C2B 2E2636FF EXP2:      LXI H,C3          ;"E" FORMAT COL ADJ
0C2F 46E800              CALL LIST
0C32 440F0C              JMP CNTU9
              ;
              ; 'SSCAN' SET SCAN ENTRY POINT. PRINTS
              ; INSTRUCTIONS AND STORES THE NUMBER
              ; AND SEQUENCE OF CHANNELS TO BE SCANNED.
              ; IT ALSO SETS THE COORDINATION NUMBER
              ; TO ZERO.
0C35 2E1036A5 SSCAN:     LXI H,STACK+5     ;RESET COORD #
0C39 3E30                MVI M,'0'
0C3B 31                  DCR L
0C3C 3E30                MVI M,'0'
0C3E 31                  DCR L
0C3F 3E30                MVI M,'0'
0C41 2E1036B0 PRM1:      LXI H,STACK+16    ;CHECK FOR WAIT FLAG
0C45 C7                  MOV A,M
0C46 3C2A                CPI '*'
0C48 485E0C              JNZ CNTU5         ;JUMP IF NOT SET
0C4B 2E1036AD            LXI H,STACK+13    ;ELSE LOAD DEFAULT
                                           ;OF 15MS
0C4F 3E01                MVI M,1
0C51 30                  INR L
0C52 3E62                MVI M,98
0C54 2E273604            LXI H,DWAIT       ;INFORM OPERATOR
0C58 46E800              CALL LIST
```

```
0C5B 467100                CALL CRLF
0C5E 2E273630 CNTU5:       LXI H,EXP          ;ASK QUESTION
0C62 46E800                CALL LIST
0C65 2E1036B1              LXI H,STACK+17     ;"E" FORMAT FLAG STORE
0C69 464500                CALL CI            ;GET VALUE
0C6C 467C00                CALL CO            ;ECHO BACK
0C6F C1                    MOV A,B            ;RESTORE A
0C70 3C4E                  CPI 'N'
0C72 687A0C                JZ NOEXP           ;JUMP IF "N"
0C75 3E59                  MVI M,'Y'          ;ELSE STORE "YES"
0C77 447C0C                JMP CNTU6
0C7A 3E4E     NOEXP:       MVI M,'N'          ;ELSE STORE "NO"
0C7C 467100   CNTU6:       CALL CRLF
0C7F 2E2636A7              LXI H,LSCAN        ;ASK QUESTION
0C83 46E800                CALL LIST
0C86 467100                CALL CRLF
0C89 2E1036C0              LXI H,SCANB        ;ENTER CHANNELS IN
                                              ; DESIRED ORDER IN
                                              ; THE SCAN BUFFER
0C8D 46FC00                CALL GETD
0C90 2E1036C0              LXI H,SCANB        ;CONVERT TO BCD
0C94 463A01                CALL STRIP
0C97 467100                CALL CRLF
0C9A 467100                CALL CRLF
0C9D 2E2636C7              LXI H,INFO         ;INSTRUCTIONS
0CA1 46E800                CALL LIST
0CA4 467100                CALL CRLF
0CA7 467100                CALL CRLF
0CAA 449700                JMP CNTRL          ;BACK TO MONITOR
              ;
              ; 'SCAN' EXECUTION POINT FOR MANUALLY SCANNING
              ; CHANNELS DEFINED ABOVE.
              ;
0CAD 2E1036C0 SCAN:        LXI H,SCANB        ;VALIDITY CHECK
0CB1 C7                    MOV A,M
0CB2 3C2A                  CPI '*'            ;BOOT DEFAULT
0CB4 68CA0C                JZ ERR1
0CB7 467100                CALL CRLF
0CBA 46340B                CALL SCAN5         ;START SCAN ROUTINE
0CBD 2E1036D0 LOOP:        LXI H,10D0H        ;WAIT FOR COMMAND
                                              ;FROM OPERATOR
0CC1 46FC00                CALL GETD
0CC4 463B0B                CALL RSCAN         ;RESCAN FOR ANOTHER
                                              ;SET OF DATA
0CC7 44BD0C                JMP LOOP           ;CONTINUE
0CCA 2E263610 ERR1:        LXI H,LERR1        ;MSG OUT
0CCE 46E800                CALL LIST
0CD1 449700                JMP CNTRL
0000           END
```

```
                ;
                ; EXTERNAL DEVICE DRIVER
                ; THIS SECTION CONTAINS THE LOGIC NECESSARY
                ; TO TURN ON TWO RELAYS ('UP' AND 'DOWN') IN
                ; ORDER TO CAUSE SOME PHYSICAL DEVICE TO
                ; MOVE TO A DESIRED LOCATION. SUBROUTINES
                ; LISTED HERE ARE ALSO USED BY THE SOFTWARE
                ; IN THE "RUN" SECTION IN ORDER TO
                ; PROVIDE AN AUTOMATIC CONTROL FUNCTION.
                ;
0000                    ORG OCEOH
                ; EQUATES NOT ANNOTATED CAN BE FOUND
                ; IN PREVIOUS SECTIONS.
                ;
10A0            STACK   EQU 10A0H
10A9            FCNT    EQU STACK+9         ;COUNTER IN NOISE
                                            ;FILTER ROUTINE
0009            CMD     EQU 8+1             ;OUTPUT PORT 1
0030            CMDUP   EQU 30H             ;ACTIVATE "UP" RELAY
0050            CMDDN   EQU 50H             ;ACTIVATE "DOWN" RELAY
0010            OFF     EQU 10H             ;RELAYS OFF
0060            BUMP    EQU 60H             ;TRANSPORT DELAY
                                            ;(SEE TEXT)
106D            RAW     EQU 106DH           ;RAW DATA INPUT BUFER
0AE9            DATA    EQU 0AE9H           ; RAW DATA TO F.P.
                                            ;LOAD ACCUM AND POINT
                                            ;TO OPERAND
006A            HALF    EQU 006AH           ;4.5 MS DELAY
0831            VRI     EQU 0831H           ;VOLT UNITS TO
                                            ;BINARY A/D COUNT
081B            RVI     EQU 081BH           ;BINARY A/D COUNT TO
                                            ;VOLTAGE UNITS
1080            CFBUF   EQU 1080H           ;CONVERSION FACTORS.
0B41            CNTU7   EQU 0B41H           ;REMOTE ENTRY TO "SCAN"
                                            ;ROUTINES WITH COLUMN
                                            ;HEADINGS OFF
10C0            SCANB   EQU 10C0H           ;BUFFER CONTAINING
                                            ;CHANNELS TO BE SCANNED
0189            FPBIN   EQU 0189H           ;BIN←F.P.
0800            SHOLD   EQU 0800H
000D            SCHAR   EQU 0DH
0065            DELAY   EQU 0065H
0071            CRLF    EQU 0071H
0097            CNTRL   EQU 0097H
00F8            GET     EQU 00F8H
013A            STRIP   EQU 013AH
0145            DISPY   EQU 0145H
00E8            LIST    EQU 00E8H
0154            BINFP   EQU 0154H
1040            DOPND   EQU 1040H
1050            FOPND   EQU 1050H
1058            STORE   EQU 1058H
033E            STR     EQU 033EH
036E            LOD     EQU 036EH
```

```
03D7           AD      EQU 03D7H
038C           MUL     EQU 038CH
03B4           DIV     EQU 03B4H
064B           INN     EQU 064BH
070F           OUU     EQU 070FH
0FE4           I5      EQU 0FE4H
               ;
               ; MESSAGES
               ;
26FF           C3      EQU 26FFH
277F           LMOVE   EQU 277FH
               ;
               ; COMMON SUBROUTINES
               ;
0CE0 C4        GT:     MOV A,E           ;RETURNS CARRY SET
                                         ;IF BC>DE
0CE1 92                SUB C
0CE2 C3                MOV A,D
0CE3 99                SBB B
0CE4 07                RET
               ;
               ;
0CE5 46EB0C    LT:     CALL SWAP         ;RETURNS CARRY SET
                                         ;IF BC<DE
0CE8 46E00C            CALL GT
               ;
               ;
0CEB C4        SWAP:   MOV A,E           ;DE←BC←DE
0CEC E2                MOV E,C
0CED D0                MOV C,A
0CEE C3                MOV A,D
0CEF D9                MOV D,B
0CF0 C8                MOV B,A
0CF1 07                RET
               ;
               ;
0CF2 3601      AB:     MVI L,01          ;ABS(ACTUAL-DESIRED)
                                         ;L=1 IF ACTUAL>
                                         ;L=0 IF ACTUAL<
0CF4 46E00C            CALL GT           ;IS DE>=BC ?
0CF7 40FE0C            JNC YES           ;JUMP IF TRUE
0CFA 31                DCR L             ;RESET FLAG
0CFB C2                MOV A,C
0CFC 94                SUB E
0CFD 07                RET
0CFE C4        YES:    MOV A,E
0CFF 92                SUB C
0D00 07                RET               ;ACTUAL>DESIRED
               ;
               ;
0D01 0610      DATA1:  MVI A,0FF         ;RELAYS OFF
0D03 2E10366E  DATA2:  LXI H,RAW+1       ;TAKE A SAMPLE FROM
                                         ;CHANNEL 0. RELAY DIR
                                         ;ASSUMED IN A
```

```
0D07 460008            CALL SHOLD
0D0A DF                MOV D,M         ;DE←SAMPLED DATA
0D0B 30                INR L
0D0C E7                MOV E,M
0D0D 31                DCR L           ;POINT TO SAMPLED DATA
0D0E 07                RET
                ;
                ; "BUMPM" IS USED TO PROVIDE AN EXTERNAL SIGNAL
                ; OF KNOWN TIME DURATION IN ORDER TO OBTAIN
                ; EXTERNAL DEVICE MOVEMENT (OPEN LOOP) OF A
                ; KNOWN AMOUNT. (SEE TEXT)
                ; INPUT: A=CHANNEL 0 PLUS RELAY DIRECTION
                ;          D=DESIRED DELAY PARAMETER
                ;
0D0F 53         BUMPM: OUT CMD         ;START RELAY
0D10 466700            CALL DELAY+2    ;KILL TIME
0D13 0610              MVI A,0FF       ;TURN RELAYS OFF
0D15 53                OUT CMD
0D16 07                RET
0D17 1E60       BUMPU: MVI D,BUMP      ;MOVE UP A SHORT DIST
                                       ;BUMP = DELAY
0D19 0630              MVI A,CMDUP
0D1B 440F0D            JMP BUMPM
0D1E 1E60       BUMPD: MVI D,BUMP      ;MOVE DOWN A SHORT DIST
0D20 0650              MVI A,CMDDN
0D22 440F0D            JMP BUMPM
                ;
                ;
0D25 3C06       ADJU:  CPI 6           ;IF LOCATION ERROR >5
                                       ;COUNTS, MOVE UP
0D27 40980D            JNC UP
0D2A 3C03              CPI 3           ;IF LOCATION ERROR > 2
                                       ;COUNTS, BUMP UP
0D2C 40170D            JNC BUMPU
0D2F 07                RET             ;CONVERGENCE EXIT
                ;
                ;
0D30 3C06       ADJD:  CPI 6           ;ERROR ADJUSTMENT (AS
                                       ;ABOVE) FOR THE DOWN
                                       ;RELAY
0D32 40B90D            JNC DOWN
0D35 3C03              CPI 3
0D37 401E0D            JNC BUMPD
0D3A 07                RET             ;CONVERGENCE EXIT
                ;
                ;
0D3B 2E64       LONG:  MVI H,100       ;1.0 SEC DELAY TO GIVE
                                       ;TIME FOR RELAY AND
                                       ;DRIVE MOTOR TO STOP
0D3D 36D0       LOOPA: MVI L,0D0H      ;INNER LOOP COUNTER
0D3F 31         LOOPB: DCR L
0D40 483F0D            JNZ LOOPB
0D43 29                DCR H
0D44 483D0D            JNZ LOOPA
```

```
0D47 07                  RET
                ;
                ; "FLTR" IS USED TO FILTER OUT "GLITCHES" FROM
                ; THE A/D CONVERTER AND TO MINIMIZE THE EFFECT
                ; OF NOISE WHICH COULD MAKE THE
                ; EXTERNALLY CONTROLLED DEVICE STOP WITH AN
                ; ABSOLUTE ERROR (ACTUAL-DESIRED) GREATER THAN
                ; AN ACCEPTABLE AMOUNT.
                ;
0D48 2E1036A9 FLTR:      LXI H,FCNT        ;N=COUNT=127
0D4C 3E7F                MVI M,127
0D4E 46010D              CALL DATA1        ;STOP AND TAKE A SAMPLE
0D51 466A00              CALL HALF         ;RELAY REACTION TIME
0D54 46EC0A              CALL DATA+3       ;CONVERT/LOAD/POINT
0D57 463E03              CALL STR          ;M←A(1)
0D5A 46010D   LOOP:      CALL DATA1        ;TAKE 127 MORE SAMPLES
                                           ;FORMING RUNNING SUM
0D5D 46EC0A              CALL DATA+3
0D60 46D703              CALL AD           ;A(N)←A(N)+M
0D63 2E103650            LXI H,FOPND
0D67 463E03              CALL STR          ;M←A(N)
0D6A 2E1036A9            LXI H,FCNT
0D6E CF                  MOV B,M           ;GET COUNTER
0D6F 09                  DCR B             ;N←N-1
0D70 F9                  MOV M,B           ;SAVE N
0D71 485A0D              JNZ LOOP          ;IF N>0, JUMP
0D74 2E103650 FAVE:      LXI H,FOPND
0D78 466E03              CALL LOD          ;A←SUM[A(N)]
0D7B 2E0F36E4            LXI H,I5
0D7F 468C03              CALL MUL          ;A←A*.0078125
0D82 2E103650            LXI H,FOPND
0D86 463E03              CALL STR          ;M←A
0D89 2E103650            LXI H,FOPND
0D8D 468901              CALL FPBIN        ;BINARY←F.P.
0D90 2E1036A7            LXI H, STACK+7    ;FETCH DESIRED ANGLE
                                           ;BC←M
0D94 CF                  MOV B,M
0D95 30                  INR L
0D96 D7                  MOV C,M
0D97 07                  RET
                ;
                ; "UP" AND "DOWN" ACTIVATE THEIR RESPECTIVE
                ; RELAYS IN ORDER TO DRIVE POSITION ERROR
                ; TO 'ZERO' (SEE TEXT)
                ;
0D98 0630     UP:        MVI A,CMDUP       ;A←UP COMMAND
0D9A 46030D              CALL DATA2        ;TAKE SAMPLE
0D9D 46E00C              CALL GT           ;IF ACTUAL<DESIRED, UP
0DA0 60980D              JC UP
0DA3 46480D              CALL FLTR         ;ELSE TURN OFF RELAY
                                           ;AND CHECK FOR NOISE
0DA6 46E00C              CALL GT           ;IF UNDERSHOOT, JUMP
0DA9 60980D              JC UP
0DAC 463B0D              CALL LONG         ;WAIT FOR DRIVE MOTOR
```

```
                                       ;TO STOP
0DAF 46F20C           CALL AB          ;COMPUTE ABSOLUTE VALUE
                                       ;OF POSITION ERROR
0DB2 31               DCR L            ;TEST FLAG
0DB3 48250D           JNZ ADJU         ;UNDERSHOOT CORRECTION
0DB6 44300D           JMP ADJD         ;OVERSHOOT CORRECTION
                                       ;EITHER ADJU OR ADJD
                                       ;TESTS FOR CONVERGENCE
              ;
              ; THE ANNOTATION FOR THE "UP" ROUTINE APPLIES
              ; ALSO TO THE FOLLOWING ROUTINE.
              ;
0DB9 0650     DOWN:   MVI A,CMDDN      ;DOWN CONTROL
0DBB 46030D           CALL DATA2
0DBE 46E50C           CALL LT
0DC1 60B90D           JC DOWN
0DC4 46480D           CALL FLTR
0DC7 46E50C           CALL LT
0DCA 60B90D           JC DOWN
0DCD 463B0D           CALL LONG
0DD0 46F20C           CALL AB
0DD3 31               DCR L
0DD4 48250D           JNZ ADJU         ;OVERSHOOT CORRECTION
0DD7 44300D           JMP ADJD         ;UNDERSHOOT CORRECTION
              ;
              ; "MOVE" SENDS A MESSAGE TO THE OPERATOR
              ; AND READS IN THE DESIRED EXTERNAL
              ; DEVICE POSITION.
              ; INPUT: UNRESTRICTED
              ; REGISTERS: ALL
              ; OUTPUT: DESIRED POSITION IS IN DE
              ;         IN A/D BINARY UNITS
              ;
0DDA 2E27367F MOVE:   LXI H,LMOVE      ;MESSAGE OUT
0DDE 46E800           CALL LIST
0DE1 467100           CALL CRLF
0DE4 46F800   MLOOP:  CALL GET         ;IN←DESIRED POSITION
                                       ;IN USER UNITS
0DE7 2E103640         LXI H,DOPND
0DEB 463A01           CALL STRIP       ;BCD←ASCII
0DEE 2E103640         LXI H,DOPND
0DF2 464B06           CALL INN         ;F.P.←BCD
0DF5 2E103680         LXI H,CFBUF      ;CONVERSION FACTOR
0DF9 46B403           CALL DIV         ;VOLTS←USER UNITS
0DFC 463808           CALL VRI+7       ;ABSOLUTE←VOLTS
0DFF 2E103640         LXI H,DOPND
0E03 463E03           CALL STR         ;M←A
0E06 2E103640         LXI H,DOPND
0E0A 468901           CALL FPBIN       ;BINARY←F.P.
0E0D 07               RET              ;BACK TO CONTROLLER
              ;
              ; "CNTUA" AND "CNTUB" ARE A CONTINUATION
              ; OF THE ABOVE SUBROUTINE FOR MANUAL CONTROL
              ; OF THE EXTERNAL DEVICE.
```

```
                  ;
0E0E 2E1036A7 CNTUA:   LXI H,STACK+7      ;M←BINARY
0E12 FB                MOV M,D
0E13 30                INR L
0E14 FC                MOV M,E
0E15 46480D            CALL FLTR          ;TAKE A SAMPLE TO
                                          ;DETERMINE DRIVE
                                          ;DIRECTION.
                                          ;(UP OR DOWN)
0E18 46E00C            CALL GT            ;TURN ON "UP" RELAY
0E1B 603D0E            JC MOVEU
0E1E 46E50C            CALL LT            ;TURN ON "DOWN" RELAY
                                          ;ELSE DO NOT MOVE
0E21 60430E            JC MOVED
0E24 2E2636FF CNTUB:   LXI H,C3           ;'    '
0E28 46E800            CALL LIST
0E2B 2E1036C0          LXI H,SCANB        ;LOAD CHANNEL 0 IN
                                          ;SCAN BUFFER
0E2F 3E00              MVI M,0
0E31 30                INR L
0E32 3E0D              MVI M,SCHAR        ;STOP AFTER 1 CHANNEL
                                          ;SCAN
0E34 46410B            CALL CNTU7         ;128 POINT AVERAGE
                                          ;WITH PREVIOUSLY
                                          ;DEFINED DELAY AND
                                          ;FORMAT. THEN PRINT
                                          ;ACTUAL POSITION
0E37 467100            CALL CRLF
0E3A 444F0E            JMP MANC1
                  ;
                  ; "MOVEU" AND "MOVED" ARE THE
                  ; ENTRY POINTS TO THE "UP" AND "DOWN"
                  ; CONTROL SUBROUTINES. (MANUAL OPS).
                  ;
0E3D 46980D       MOVEU:  CALL UP
0E40 44240E               JMP CNTUB
0E43 46B90D       MOVED:  CALL DOWN
0E46 44240E               JMP CNTUB
                  ;
                  ; "MANC" IS THE ENTRY POINT FOR THE
                  ; OPERATOR FOR MANUAL ACTUATION OF SOME
                  ; MICROPROCESSOR CONTROLLED DEVICE.
                  ;
0E49 46DA0D       MANC:   CALL MOVE
0E4C 440E0E               JMP CNTUA
0E4F 46E40D       MANC1:  CALL MLOOP
0E52 440E0E               JMP CNTUA
0000               END
```

```
                 ; DIAGNOSTICS
                 ; "DUMP" IS USED TO DISPLAY THE CONTENTS
                 ; OF THE CONVERSION FACTOR BUFFER. "TEST"
                 ; CHECKS ALL RAM BETWEEN 1000H AND 1FFFH BY
                 ; WRITING OUT A BYTE TO EACH LOCATION,
                 ; READING IT BACK, AND COMPARING TO THE
                 ; ORIGINAL VALUE. IF AN ERROR IS DECTECTED,
                 ; THE TTY BELL IS RUNG AND A MESSAGE IS
                 ; PRINTED OUT ALONG WITH THE CONTENTS OF THE
                 ; BAD MEMORY LOCATION AND IT'S ADDRESS.
                 ;
0000                     ORG 0E60H
                 ; EQUATES NOT ANNOTATED CAN BE FOUND
                 ; IN PREVIOUS SECTIONS.
                 ;
0080             CFB     EQU 080H           ;LOW ADD OF CONVERSION
                                            ;FACTOR BUFFER
00A0             STK     EQU 0A0H           ;LOW ADDRESS OF STACK
007C             CO      EQU 007CH
10A0             STACK   EQU 10A0H
0071             CRLF    EQU 0071H
00E8             LIST    EQU 00E8H
0145             DISPY   EQU 0145H
0097             CNTRL   EQU 0097H
1070             RESLT   EQU 1070H
036E             LOD     EQU 036EH
070F             OUU     EQU 070FH
                 ;
                 ; MESSAGES
                 ;
27A5             RAM     EQU 27A5H
                 ;
0E60 2E1036A0    DUMP:   LXI H,STACK
0E64 3E80                MVI M,CFB          ;STARTING LOCATION
0E66 F7          DLOOP:  MOV L,M            ;GET VECTOR
0E67 466E03              CALL LOD           ;A←CONVERSION FACTOR
0E6A 2E103670            LXI H,RESLT
0E6E 460F07              CALL OUU           ;DUMP TO OUTPUT
                                            ;BUFFER
0E71 464501              CALL DISPY         ;ASCII←BCD
0E74 2E103670            LXI H,RESLT
0E78 46E800              CALL LIST          ;PRINTOUT INFO
0E7B 467100              CALL CRLF          ;NEW LINE
0E7E 2E1036A0            LXI H,STACK
0E82 C7                  MOV A,M            ;FETCH VECTOR
0E83 0404                ADI 4              ;A←A+4
0E85 F8                  MOV M,A            ;SAVE VECTOR
0E86 3CA0                CPI STK            ;CHECK FOR LAST
0E88 689700              JZ CNTRL           ;IF TRUE, DONE
0E8B 44660E              JMP DLOOP          ;ELSE GET NEXT
                 ;
                 ;
                 ;
```

```
                ;
                ;
                ; "MTEST" IS USED TO CHECK EACH RAM LOCATION
                ; TO ENSURE IT IS ALL IN WORKING ORDER. IT
                ; ALTERNATELY WRITES ALL 0'S THEN ALL
                ; 1'S TO EACH CELL AND TESTS THAT THE VALUE
                ; READ BACK IS THE ONE IT SENT OUT.
0E8E 0600       MTEST:  MVI A,0         ;FIRST TEST VALUE
0E90 2E103600   NEXT:   LXI H,1000H     ;RAM START LOCATION
0E94 C8         LOOP:   MOV B,A         ;SAVE A
0E95 F8                 MOV M,A         ;WRITE TEST VALUE
0E96 C7                 MOV A,M         ;READ TEST VALUE
0E97 B9                 CMP B           ;IS IT THE SAME ?
0E98 48B60E             JNZ ERR3        ;IF NO JUMP
0E9B 30                 INR L           ;POINT TO NEXT
0E9C 68A20E             JZ PC           ;CHECK PAGE CROSSING
0E9F 44940E             JMP LOOP        ;TRY ANOTHER
0EA2 28         PC:     INR H           ;NEXT PAGE
0EA3 C5                 MOV A,H
0EA4 3C20               CPI 20H         ;LAST PAGE ?
0EA6 68AD0E             JZ NEW          ;IF YES, CHECK LAST
                                        ;VALUE
0EA9 C1                 MOV A,B         ;RESTORE
0EAA 44940E             JMP LOOP
0EAD C1         NEW:    MOV A,B         ;GET CURRENT TEST
                                        ;VALUE
0EAE 3CFF               CPI 0FFH        ;IS IT FFH
0EB0 2B                 RZ              ;IF TRUE, DONE
0EB1 06FF               MVI A,0FFH      ;ELSE SET NEW VALUE
0EB3 44900E             JMP NEXT        ;RETEST RAM WITH
                                        ;NEW VALUE
0EB6 46D10E     ERR3:   CALL HEXT       ;PRINT OUT BAD DATA
0EB9 0E2F               MVI B,'/'
0EBB 467C00             CALL CO
0EBE C5                 MOV A,H
0EBF 46D10E             CALL HEXT       ;ADDRESS OUT
0EC2 C6                 MOV A,L
0EC3 46D10E             CALL HEXT
0EC6 467100             CALL CRLF
0EC9 2E2736A5           LXI H,RAM       ;TELL OPERATOR
0ECD 46E800             CALL LIST
0ED0 07                 RET             ;DONE
                ;
                ; "HEXT" OUTPUTS HEX DATA IN ACCUM AS
                ; TWO ASCII DIGITS.
                ;
0ED1 E0         HEXT:   MOV E,A         ;SAVE
0ED2 1A                 RAR             ;LO NIBBLE←HI NIBBLE
0ED3 1A                 RAR
0ED4 1A                 RAR
0ED5 1A                 RAR
0ED6 46E40E             CALL HEX        ;OUTPUT ROUTINE
0ED9 467C00             CALL CO
0EDC C4                 MOV A,E         ;RESTORE
0EDD 46E40E             CALL HEX
```

```
0EE0 467C00             CALL CO
0EE3 07                 RET
0EE4 240F      HEX:     ANI 0FH         ;MASK OFF HI NIBBLE
0EE6 3C0A               CPI 10          ;IS IT A NUMBER
0EE8 60ED0E             JC NUM          ;IF TRUE, JUMP
0EEB 0407               ADI 7           ;ELSE CONSTRUCT LETTER
0EED 0430      NUM:     ADI 30H         ;ASCII BIAS
0EEF C8                 MOV B,A         ;OUTPUT REGISTER
0EF0 07                 RET
0000            END
```

```
                ;
                ; THE AUTMATIC CONTROL SECTION MAKES USE OF
                ; THE "MOVE" AND "SCAN" SECTIONS TO PROVIDE
                ; AUTOMATIC,INCREMENTAL STEPPING OF AN
                ; EXTERNAL DEVICE BETWEEN ARBITRARY LIMITS.
                ;
0000                    ORG 2000H
                ; EQUATES NOT ANNOTATED CAN BE FOUND
                ; IN PREVIOUS SECTIONS
                ;
0CEB            SWAP    EQU 0CEBH       ;DE←BC←DE
0CE0            GT      EQU 0CE0H       ;DE-BC,RES NOT SAVED
0CE5            LT      EQU 0CE5H       ;BC-DE,RES NOT SAVED
0DE4            MLOOP   EQU 0DE4H       ;FETCH EXTERNAL
                                        ;POSIT,CONVERT, AND
                                        ;STORE IN DE
0D48            FLTR    EQU 0D48H       ;GLITCH AND NOISE
                                        ;FILTER
0B3B            RSCAN   EQU 0B3BH       ;RE-SCAN DESIRED
                                        ;CHANNELS AND PRINT
                                        ;RESULTS
0B34            SCAN5   EQU 0B34H       ;PRINT OUT
                                        ;COLUMN HEADINGS
0CCA            ERR1    EQU 0CCAH       ;TERMINAL ERROR
0D98            UP      EQU 0D98H       ;TURN ON UP DRIVE
0DB9            DOWN    EQU 0DB9H       ;TURN ON DOWN DRIVE
1060            N       EQU 1060H       ;ITERATION COUNTER
                                        ;FLOATING POINT
                                        ;STORAGE
1064            TEMP    EQU 1064H       ;TEMPORARY PRODUCT
                                        ;STORAGE
1068            FINC    EQU 1068H       ;F.P. REPRESENTATION
                                        ;OF INCREMENTAL DIST
0FE0            I6      EQU 0FE0H       ;1.0
0FF0            I1      EQU 0FF0H       ;819.15
000D            SCHAR   EQU 0DH
0071            CRLF    EQU 0071H
00E8            LIST    EQU 00E8H
0097            CNTRL   EQU 0097H
00F8            GET     EQU 00F8H
013A            STRIP   EQU 013AH
0154            BINFP   EQU 0154H
0189            FPBIN   EQU 0189H
1040            DOPND   EQU 1040H
1058            STORE   EQU 1058H
10A0            STACK   EQU 10A0H
10C0            SCANB   EQU 10C0H
1080            CFBUF   EQU 1080H
0080            CFB     EQU 080H
033E            STR     EQU 033EH
036E            LOD     EQU 036EH
03D7            AD      EQU 03D7H
038C            MUL     EQU 038CH
03B4            DIV     EQU 03B4H
```

```
064B             INN     EQU 064BH
                 ;
                 ; MESSAGES
                 ;
2672             LREAD   EQU 2672H
2746             START   EQU 2746H
2755             STOP    EQU 2755H
2764             INCRE   EQU 2764H
2772             LUNIT   EQU 2772H
                 ;
                 ; COMMON SUBROUTINES
                 ;
                 ; "LODXX" AND "STRXX" ARE USED FOR 2 WAY
                 ; TRANSFER OF DATA BETWEEN CPU
                 ; REGISTERS AND MEMORY.
                 ;
2000 2E1036A7 STRD:    LXI H,STACK+7    ;CURRENT DESIRED
                                        ;POSITION STORAGE
2004 F9       MBC:     MOV M,B
2005 30                INR L
2006 FA                MOV M,C
2007 07                RET
2008 2E1036A7 LODD:    LXI H,STACK+7    ;BC←M
200C CF       BCM:     MOV B,M
200D 30                INR L
200E D7                MOV C,M
200F 07                RET
2010 2E1036B2 STRST:   LXI H,STACK+18   ;START POSITION
2014 440420            JMP MBC          ;M←BC
2017 2E1036B2 LODST:   LXI H,STACK+18   ;BC←M
201B 440C20            JMP BCM
201E 2E1036B4 STRS:    LXI H,STACK+20   ;STOP POSITION
2022 FB       MDE:     MOV M,D
2023 30                INR L
2024 FC                MOV M,E
2025 07                RET
2026 2E1036B4 LODS:    LXI H,STACK+20   ;DE←M
202A DF       DEM:     MOV D,M
202B 30                INR L
202C E7                MOV E,M
202D 07                RET
202E 2E1036B6 STRI:    LXI H,STACK+22   ;INCREMENTAL POSITION
2032 442220            JMP MDE
2035 2E1036B6 LODI:    LXI H,STACK+22   ;DE←M
2039 442A20            JMP DEM
              ;
              ;
203C 2E103660 INCN:    LXI H,N          ;INCREMENT ITERATION
                                        ;COUNTER  (N)
2040 466E03            CALL LOD         ;A←N
2043 2E0F36E0          LXI H,I6
2047 46D703            CALL AD          ;A=A+1
204A 2E103660          LXI H,N
204E 463E03            CALL STR         ;N←A
```

```
2051 2E103668         LXI H,FINC      ;GET INCREMENT (I)
2055 468C03           CALL MUL        ;I←N*I
2058 2E103664         LXI H,TEMP      ;SAVE FACTOR
205C 463E03           CALL STR
205F 2E103664         LXI H,TEMP
2063 468901           CALL FPBIN      ;BINARY←F.P.
2066 462E20           CALL STRI       ;M←NEW INCREMENT
2069 463520   LOAD:   CALL LODI       ;LOAD START POSIT
                                      ;AND INCREMENT
206C 441720           JMP LODST
206F 460020   STOR:   CALL STRD       ;STORE NEXT POSIT
2072 442620           JMP LODS        ;LOAD STOP POSIT
              ;
              ;
              ; "INCP" AND "DECP" ARE USED TO INCREMENT/
              ; DECREMENT THE VALUE OF "DESIRED POSITION"
              ; BY "INCREMENT" AMOUNT.
              ; THE ROUTINE RETURNS WITH:
              ;  BC←START POSIT+INCREMENT*N
              ;  STACK+7←BC
              ;  DE←STOP POSIT
              ;
2075 463C20   INCP:   CALL INCN       ;INPUT PARAMETERS
2078 C4               MOV A,E         ;ADD BC+DE AND STORE
                                      ;RESULT IN BC
2079 82               ADD C
207A D0               MOV C,A
207B C3               MOV A,D
207C 89               ADC B
207D C8               MOV B,A
207E 446F20           JMP STOR        ;EXIT THROUGH STORE
2081 463C20   DECP:   CALL INCN       ;INPUT PARAMETERS
2084 C2               MOV A,C         ;SUB DE-BC AND STORE
                                      ;RESULT IN BC
2085 94               SUB E
2086 D0               MOV C,A
2087 C1               MOV A,B
2088 9B               SBB D
2089 C8               MOV B,A
208A 446F20           JMP STOR        ;EXIT
              ;
              ;
208D 2E1036B8 SSUB:   LXI H,STACK +24 ;SET SUBT FLAG
2091 3E01             MVI M,1
2093 441B21           JMP RUNL        ;BACK TO "RUN LOOP"
              ;
              ; "RUN" IS THE OPERATOR ENTRY POINT TO
              ; TO THE AUTOMATIC CONTROL ROUTINE. IT
              ; PROVIDES REPEATED SCANNING OF UP TO
              ; 8 CHANNELS AT SELECTED POSITIONS OF AN
              ; EXTERNAL DEVICE.
              ;
2096 2E1036C0 RUN:    LXI H,SCANB     ;VALIDITY CHECK
209A C7               MOV A,M
```

```
209B 3C2A               CPI '*'           ;BOOT DEFAULT
209D 68CA0C             JZ ERR1
20A0 2E273646           LXI H,START       ;GET START POSITION
20A4 46E800             CALL LIST
20A7 46E40D             CALL MLOOP        ;CONVERT TO BIN
20AA 46EB0C             CALL SWAP         ;BC←DE
20AD 460020             CALL STRD         ;M←START
20B0 461020             CALL STRST        ;M←START
20B3 467100             CALL CRLF
20B6 2E273655           LXI H,STOP        ;GET FINAL POSITION
20BA 46E800             CALL LIST
20BD 46E40D             CALL MLOOP        ;CONVERT TO BIN
20C0 461E20             CALL STRS         ;M←STOP
20C3 467100             CALL CRLF
20C6 2E273664           LXI H,INCRE       ;GET INCREMENT
20CA 46E800             CALL LIST
20CD 46F800             CALL GET          ;INPUT←I
20D0 2E103640           LXI H,DOPND
20D4 463A01             CALL STRIP        ;BCD←ASCII
20D7 2E103640           LXI H,DOPND
20DB 464B06             CALL INN          ;F.P.←BCD
20DE 2E103680           LXI H,CFBUF
20E2 46B403             CALL DIV          ;I←VOLT(I)
20E5 2E0F36F0           LXI H,I1
20E9 468C03             CALL MUL          ;I←BIN(I)
20EC 2E103668           LXI H,FINC
20F0 463E03             CALL STR          ;M←I
20F3 2E103668           LXI H,FINC
20F7 468901             CALL FPBIN        ;DE←ABSOLUTE(M)
20FA 462E20             CALL STRI         ;M←INCREMENT
20FD 2E103660           LXI H,N           ;RESET COUNTER
2101 3E00               MVI M,0
                ;
2103 467100             CALL CRLF
2106 46340B             CALL SCAN5        ;COL HEADINGS
2109 460820             CALL LODD         ;GET START AND STOP
                                          ;POSIT TO DETERMINE
                                          ;INCREMENT DIRECTION
210C 462620             CALL LODS
210F 46E00C             CALL GT           ;START>STOP ?
2112 608D20             JC SSUB           ;IF YES, SET FLAG
2115 2E1036B8           LXI H,STACK+24    ;ELSE RESET FLAG
2119 3E00               MVI M,0
211B 46480D     RUNL:   CALL FLTR         ;TAKE POSIT READINGS
211E 46E00C             CALL GT
2121 604C21             JC INCR           ;IF DESIRED>ACTUAL,
                                          ;MOVE UP
2124 46E50C             CALL LT
2127 605221             JC DECR           ;IF DESIRED<ACTUAL,
                                          ;MOVE DOWN
212A 463B0B     CNTUC:  CALL RSCAN        ;ELSE TAKE A SET OF
                                          ;CHANNEL READINGS
212D 2E1036B8           LXI H,STACK+24    ;GET DIRECTION FLAG
2131 C7                 MOV A,M
```

```
2132 3C01               CPI 1
2134 684321             JZ DECR1            ;IF SET, DECREASE
                                            ;POSIT BY "I"
2137 467520    INCR1:   CALL INCP           ;ELSE INCREASE POSIT
213A 46E00C             CALL GT
213D 605821    TEST:    JC EXIT             ;IF STOP POSIT
                                            ;EXCEEDED, EXIT
2140 441B21             JMP RUNL            ;ELSE REPEAT
2143 468120    DECR1:   CALL DECP
2146 46E50C             CALL LT
2149 443D21             JMP TEST            ;CHECK FOR STOP
                                            ;POSIT EXCEEDED
214C 46980D    INCR:    CALL UP             ;MOVE UP
214F 442A21             JMP CNTUC
2152 46B90D    DECR:    CALL DOWN           ;MOVE DOWN
2155 442A21             JMP CNTUC
2158 467100    EXIT:    CALL CRLF
215B 467100             CALL CRLF
215E 449700             JMP CNTRL
               ;
               ;
               ; "UNIT" IS USED TO INPUT CONVERSION FACTORS
               ; WHICH CHANGE THE INTERNAL UNITS (VOLTS)
               ; TO ANY UNIT DEFINED BY THE USER. ALL I/O
               ; OPERATION IS THEN IN TERMS OF THESE
               ; NEW UNITS UNTIL RESET.
               ;
2161 2E263672  UNIT:    LXI H,LREAD         ;"CHANNEL = ?"
2165 46E800             CALL LIST
2168 46F800             CALL GET            ;INPUT CHANNEL
216B 2E103640           LXI H,DOPND
216F 463A01             CALL STRIP          ;BCD←ASCII
2172 2E103640           LXI H,DOPND
2176 C7                 MOV A,M             ;GET CHANNEL
2177 B0                 ORA A               ;CLEAR CARRY
2178 12                 RAL                 ;MPY BY 4
2179 12                 RAL
217A 0480               ADI CFB             ;COMPUTE VECTOR
217C 2E1036A6           LXI H,STACK+6
2180 F8                 MOV M,A             ;STORE IT
2181 467100             CALL CRLF
2184 2E273672           LXI H,LUNIT         ;"UNIT/VOLT =?"
2188 46E800             CALL LIST
218B 46F800             CALL GET            ;GET CONVERSION FACTOR
218E 2E103640           LXI H,DOPND
2192 463A01             CALL STRIP          ;BCD←ASCII
2195 2E103640           LXI H,DOPND
2199 464B06             CALL INN            ;F.P.←BCD
219C 2E1036A6           LXI H,STACK+6       ;GET VECTOR
21A0 F7                 MOV L,M             ;POINT TO STORAGE
21A1 463E03             CALL STR            ;M←FACTOR
21A4 467100             CALL CRLF
21A7 446121             JMP UNIT            ;GET NEXT
0000                END
```

```
                   ;
                   ; ALL MESSAGES USED BY THE SYSTEM ARE
                   ; CONTAINTED IN THIS SECTION.
                   ;
0000                        ORG 2600H
                   ;
000D               SCHAR    EQU 0DH             ;STOP CHARACTER
000A               LF       EQU 0AH             ;END OF RECORD
                   ;
2600 3E200D        READY:   DB '> ',SCHAR
2603 204E4F54      LERR:    DB ' NOT DEFINED',SCHAR
2607 20444546
260B 494E4544
260F 0D
2610 30313A20      LERR1:   DB '01: CHANNELS '
2614 4348414E
2618 4E454C53
261C 20
261D 4E4F5420               DB 'NOT DEFINED',SCHAR
2621 44454649
2625 4E45440D
2629 30323A20      LERR2:   DB '02: INVALID '
262D 494E5641
2631 4C494420
2635 2246494C               DB '"FILE" '
2639 452220
263C 5445524D               DB 'TERMINATION',SCHAR
2640 494E4154
2644 494F4E0D
2648 2A2A2A20      LBOOT:   DB '*** RESET: ALL '
264C 52455345
2650 543A2041
2654 4C4C20
2657 4348414E               DB 'CHANNEL I/O IN '
265B 4E454C20
265F 492F4F20
2663 494E20
2666 22564F4C               DB '"VOLTS" ***',SCHAR
266A 54532220
266E 2A2A2A0D
2672 20434841      LREAD:   DB ' CHANNEL = ',SCHAR
2676 4E4E454C
267A 203D200D
267E 56414C49      LWAIT:   DB 'VALID FACTORS: ',SCHAR
2682 44204641
2686 43544F52
268A 533A200D
268E 41203D20               DB 'A = 3MS',SCHAR
2692 334D530D
2696 42203D31               DB 'B =15MS',SCHAR
269A 354D530D
269E 43203D32               DB 'C =25MS',LF,SCHAR
26A2 354D530A
26A6 0D
```

```
26A7 494E5055 LSCAN:  DB 'INPUT CHANNELS'
26AB 54204348
26AF 414E4E45
26B3 4C53
26B5 20494E20         DB ' IN DESIRED '
26B9 44455349
26BD 52454420
26C1 4F524445         DB 'ORDER',SCHAR
26C5 520D
26C7 5748454E INFO:   DB 'WHEN READY TO '
26CB 20524541
26CF 44592054
26D3 4F20
26D5 54414B45         DB 'TAKE DATA, '
26D9 20444154
26DD 412C20
26E0 54595045         DB 'TYPE  SCAN '
26E4 20205343
26E8 414E20
26EB 4F522052         DB 'OR RUN ',SCHAR
26EF 554E200D
26F3 2023200D C1:     DB ' # ',SCHAR
26F7 20202043 C2:     DB '   CH. ',SCHAR
26FB 482E200D
26FF 20202020 C3:     DB '    ',SCHAR
2703 0D
2704 44454C41 DWAIT:  DB 'DELAY BETWEEN '
2708 59204245
270C 54574545
2710 4E20
2712 44415441         DB 'DATA POINTS = '
2716 20504F49
271A 4E545320
271E 3D20
2720 3135204D         DB '15 MS (DEFAULT)',SCHAR
2724 53202844
2728 45464155
272C 4C54290D
2730 22452220 EXP:    DB '"E" FORMAT'
2734 464F524D
2738 4154
273A 2859204F         DB '(Y OR N) ? ',SCHAR
273E 52204E29
2742 203F200D
2746 53544152 START:  DB 'START POSIT = ',SCHAR
274A 5420504F
274E 53495420
2752 3D200D
2755 53544F50 STOP:   DB 'STOP POSIT  = ',SCHAR
2759 20504F53
275D 49542020
2761 3D200D
```

```
2764 494E4352 INCRE:  DB 'INCREMENT  = ',SCHAR
2768 454D454E
276C 5420203D
2770 200D
2772 554E4954 LUNIT:  DB 'UNIT/VOLT = ',SCHAR
2776 2F564F4C
277A 54203D20
277E 0D
277F 44455349 LMOVE:  DB 'DESIRED .... '
2783 52454420
2787 2E2E2E2E
278B 20
278C 41435455         DB 'ACTUAL',SCHAR
2790 414C0D
2793 52554E20 LRUN:   DB 'RUN NO. ',SCHAR
2797 4E4F2E20
279B 0D
279C 2A2A2A2A LCOM:   DB '*****   ',SCHAR
27A0 2A202020
27A4 0D
27A5 07444154 RAM:    DB 07H,'DATA/'
27A9 412F
27AB 4C4F4341         DB 'LOCATION'
27AF 54494F4E
27B3 202E2E2E         DB ' ...... BAD RAM'
27B7 2E2E2E20
27BB 42414420
27BF 52414D
27C2 070D             DB 07H,SCHAR
0000                END
```

```
                ;
                ;
                ; JUMP TABLE
                ;
                ; THE RECOGNITION ROUTINE ("RECOG", 00A1H)
                ; COMPARES THE CHECK-SUM IT COMPUTES WITH
                ; EVERY THIRD ENTRY IN THIS TABLE. IF A
                ; MATCH IS FOUND, THE FOLLOWING TWO BYTES
                ; SHOW THE ENTRY POINT FOR THE DESIRED
                ; ROUTINE.
                ;
                ; 00H MARKS THE END OF THE TABLE.
                ;
0000                    ORG 0F00H
                ;
                ; EQUATES
                ;
0847            READ    EQU 0847H       ;VOLTMETER FUNCTION
08F0            CNTLR   EQU 08F0H       ;"RUN NO. "
0901            CNTLC   EQU 0901H       ;"*****  " COMMEMT
0912            EDIT    EQU 0912H       ;TEXT INPUT
095D            FILE    EQU 095DH       ;WRITE TEXT
0963            WAIT    EQU 0963H       ;DELAY FACTOR
098B            A1      EQU 098BH       ;3 MS DELAY
097F            B1      EQU 097FH       ;15 MS DELAY
0973            C1      EQU 0973H       ;25 MS DELAY
0C35            SSCAN   EQU 0C35H       ;SET SCAN ROUTINE
0CAD            SCAN    EQU 0CADH       ;TAKE DATA
0E49            MOVE    EQU 0E49H       ;MANUAL CONTROL
0E60            DUMP    EQU 0E60H       ;CONVERSION FACTORS
0E8E            MTEST   EQU 0E8EH       ;RAM CHECK
2096            RUN     EQU 2096H       ;AUTOMATIC CONTROL
2161            UNIT    EQU 2161H       ;INPUT SCALE FACTORS
                ;
0F00 1C         J1:     DB 1CH
0F01 4708               DW READ
0F03 12         J2:     DB 12H
0F04 F008               DW CNTLR
0F06 03         J3:     DB 03H
0F07 0109               DW CNTLC
0F09 26         J4:     DB 26H
0F0A 1209               DW EDIT
0F0C 20         J5:     DB 20H
0F0D 5D09               DW FILE
0F0F 35         J6:     DB 35H
0F10 6309               DW WAIT
0F12 41         J7:     DB 41H
0F13 8B09               DW A1
0F15 42         J8:     DB 42H
0F16 7F09               DW B1
0F18 43         J9:     DB 43H
0F19 7309               DW C1
0F1B 31         J10:    DB 31H
0F1C 350C               DW SSCAN
```

```
0F1E 25        J11:    DB 25H
0F1F AD0C              DW SCAN
0F21 37        J12:    DB 37H
0F22 490E              DW MOVE
0F24 36        J13:    DB 36H
0F25 600E              DW DUMP
0F27 8D        J14:    DB 8DH
0F28 8E0E              DW MTEST
0F2A F5        J15:    DB 0F5H
0F2B 9620              DW RUN
0F2D 40        J16:    DB 40H
0F2E 6121              DW UNIT
0F30 00        STOP:   DB 00H
```

```
              ;
              ; CONSTANT STORAGE
              ;
              ; THE FOLLOING DATA ARE THE FLOATING
              ; POINT REPRESENTATION OF CONSTANTS
              ; USED THROUGHOUT THE PROGRAM
              ;
0F31                  ORG 0FE0H
0FE0 81000000 I6:     DB 81H,0,0,0      ;1.0
0FE4 7A000000 I5:     DB 7AH,0,0,0      ;.0078125
              ;
0FE8                  ORG 0FF0H
0FF0 8A4C     I1:     DB 8AH,4CH
0FF2 C99A             DB 0C9H,9AH       ;819.15
0FF4 84200000 I2:     DB 84H,20H,0,0    ;10.0
0FF8 8A4C     I3:     DB 8AH,4CH
0FFA C99A             DB 0C9H,9AH       ;819.15
0FFC 8D7F     I4:     DB 8DH,7FH
0FFE FC00             DB 0FCH,0         ;8191.15
0000           END
```

VI. RECOMMENDATIONS

The ADL software was developed on an existing development system which used the following:

1. 110 baud teletype for program listing.
2. 110 baud paper tape punch for mass storage.
3. 1200 baud high speed paper tape reader.
4. 1200 baud CRT for program entry and editing.

While this system is a useful tool for writing and debugging small programs, it is not a viable system for large scale development. The percentages of time devoted to the creation of the ADL software package was 15% logic development, 5% manual entry, 15% debugging and 75% waiting for paper tape and teletype output. The last figure represents a significant and costly waste of manpower assets. The following system - while more expensive - could easily pay for itself in man-hour savings alone:

1. Floppy disk mass memory. this reduces an edit-assembly-reedit-reassembly cycle from up to eight hours (for the entire package) to less than five minutes (also for the entire package).
2. Line printer for producing source code and assembly listings.
3. Resident high level language such as BASIC or PL/M to enhance complex logic manipulations.

The Department of Aeronautics has recently acquired the

INTEL MDS 80 development system. This system contains the abcve components and is presently being used as a data acquisition system for an oscillating flow wind tunnel. In addition to data logging, this system can perform on-line fast fourier analysis of data taken in a highly turbulent and non-linear environment [2].

Microprocessor usage presents a unique problem; namely, better CPUs and more advanced peripherals appear on the market almost monthly. Therefore, a U-P oriented system rapidly becomes outdated. The software for the ADL was written using industry standard techniques. A change to the more advanced 8080 CPU can therefore be accomplished (with minor changes) by simply reassembling the program with an 8080 assembler. Such an update is recommended if the ADL is to be used to take higher frequency data.

APPENDIX A

GLOSSARY

1. A/D: analog to digital (adjective or noun)

2. assembly: A listing which contains both source code and machine code.

3. BAUD: A data transmission rate expressed in BITs per second.

4. BIT: BInary digiT. A single unit of information in a binary word.

5. buffer: A group of memory locations used to store specific data (input data, constants, output data, etc.).

6. buffering: A process by which electronic signals possessing different properties are made compatible.

7. byte: An eight-BIT word which is processed as a single quantity.

8. CPU: Central Processing Unit. The area of the microprocessor which computes and sequences all logic and arithmetic functions.

9. coordination number: A sequential, numerical label associated with a set of data points for a given run.

10. CRT: Cathod Ray Tube. Also used as the generic name for a television type display.

11. D/A: The inverse of the A/D process.

12. data logging: The acquisition and tabulation of data.

13. EPROM: erasable/programmable read only memory

14. driver: In a software context this term refers to a program used to control the actions of an external device.

15. external device: A physical device which is not an integral part of the microprocessor.

16. glitch: A missing BIT in a byte of data which can occur during data transmission or conversion.

17. H: A suffix which indicates a hexadecimal number (Appendix C).

18. I/O: input/output

19. K: A suffix which indicates a group of 1024 (2^{10}) items as in '4K of memory' meaning 4096 memory locations.

20. machine code: The BIT patterns actually used by the U-P in order to carry out its assigned logic functions.

21. MUX: a multiplexing device

22. nibble: The upper or lower four BITs in one byte.

23. OS: Operating System. Another term for Software Package.

24. page: a 256 byte segment of memory

25. RAM: Random access memory. Volatile memory used for variable storage and data manipulation.

26. register: A storage location located in the CPU.

27. ROM: read only memory, non-volatile

28. software: The program which resides in the U-P's memory.

29. source code: The program written by the user.

30. U-P: microprocessor

31. 8008: An 8-BIT U-P device.

32. 8080: The next generation U-P from the 8008.

APPENDIX B

VENDOR DATA

The following specification sheets give the major properties of the hardware used in the ADL system. Also presented are the I/O pin assignments for the 805 processor as well as the pin-outs for the other connecters used throughout the system.

MPS 805 MICROPROCESSOR SYSTEM SPECIFICATIONS

Physical

Three 4.5" by 6.5" printed circuit cards
- One 8111 CPU card
- One 8114 Input card
- One 8115 Output card
- One 8116 ROM card
- One 8117 RAM card

Connector Requirement for each card
56 pin, 28 position dual read-out on 0.125 centers

CPU Card includes
8008 CPU
Crystal clock
Address latches, data buffers, and control decode circuits.
Power-on and external restart.
DMA buffers

ROM Card includes
One 1702A PROM (256 bytes) and eight PROM sockets
Socket for card expansion circuit (up to 8 cards)

RAM Card includes
Eight 2102 RAM (1024 bytes) and thirty-two RAM sockets
Socket for card expansion circuit (up to 4 cards)

Input Card includes
32 TTL input selector circuits addressable in groups of 8
Socket for card expansion circuit (up to 2 cards)

Output Card includes
32 TTL output latch circuits addressable in groups of 8
Socket for card expansion circuit (up to 6 cards).

Operational

CPU
Executes all of the 8008 instructions.
4 microsecond time state cycle using 8008 (MPS 1805).
2.8 microsecond time state cycle using 8008-1 (MPS 805-1).

Memory for data or program storage card expandable to any combination of ROM and RAM to 16384 words
ROM, 2048 word capacity per card.
RAM, 4096 word capacity per card

Input and Output
Input gates implement the INP instructions.
Output latches implement the OUT instructions.

Interrupt or External Restart
Single line, synchronized interrupt on CPU card can be optionally wired for multi-level interrupt or Power-on external restart.
Multi-level Interrupt. Control lines available for external interrupt such as 8118 priority interrupt card.
Power-on and external restart option. CPU starts at instruction location 0000 by wiring restart output from CPU card to Interrupt Request input

DMA (Direct Memory Access)
Data, address, and control lines are 3-state disconnected by the CPU following a HLT instruction allowing DMA by peripherals. The CPU must be interrupted to continue following a HALT

Electrical Requirements
Refer to individual data sheets and schematics on the 8111, 8114, 8115, 8116, and 8117 for interface and wiring.

Power Requirements for the five card set fully loaded
+VCC = ±5% @ 3.3 Amp maximum (35mA per ROM, 50mA per RAM)
GND 0 volts
VDD = 9 volts ±5% @ 900 mA maximum (35 mA per ROM)

Hardware
Compatible with Series 8400 interface cards.
Fits CR5, CR10 or CR19 card racks
Use M273 power supply
PROM's programmable on Series 81 programmers

Software
MPS 800 hardware is fully compatible with any 8008 software assuming I/O and interrupt can be assigned compatibly. Teletype operating system and system monitor available. Assemblers, compilers and simulators available through computer time-sharing services

8 CHANNEL ANALOG MULTIPLEXER

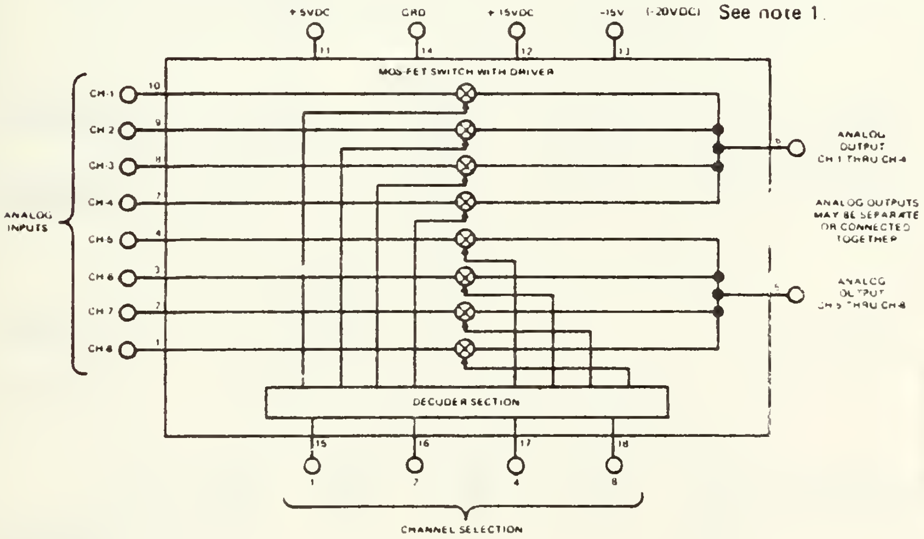
MODEL MM-8

FOR ANALOG TIME SHARING - $69 each

FEATURES

□ Small size	1" x 2" x 0.375"
□ Low power consumption	300 milliwatts
□ High transfer accuracy	±0.01%
□ Fast settling output	1 microsecond to ±0.01% of FS.
□ Choice of input type	Single ended or differential
□ Completely self contained ...	Includes 8 MOS-FET switches, drivers and decoding logic for channel selection

BLOCK DIAGRAM


+5VDC
GRD
+15VDC
-15V
(-20VDC)
11
14
12
13
MOS-FET SWITCH WITH DRIVER
CH-1
CH 2
CH-3
CH-4
CH-5
CH 6
CH 7
CH 8
10
9
8
7
4
3
2
1
ANALOG INPUTS
ANALOG OUTPUT CH 1 THRU CH-4
ANALOG OUTPUTS MAY BE SEPARATE OR CONNECTED TOGETHER
ANALOG OUTPUT CH 5 THRU CH-8
6
5
DECODER SECTION
15
16
17
18
1
2
4
8
CHANNEL SELECTION

See note 1.

SAMPLE AND HOLD

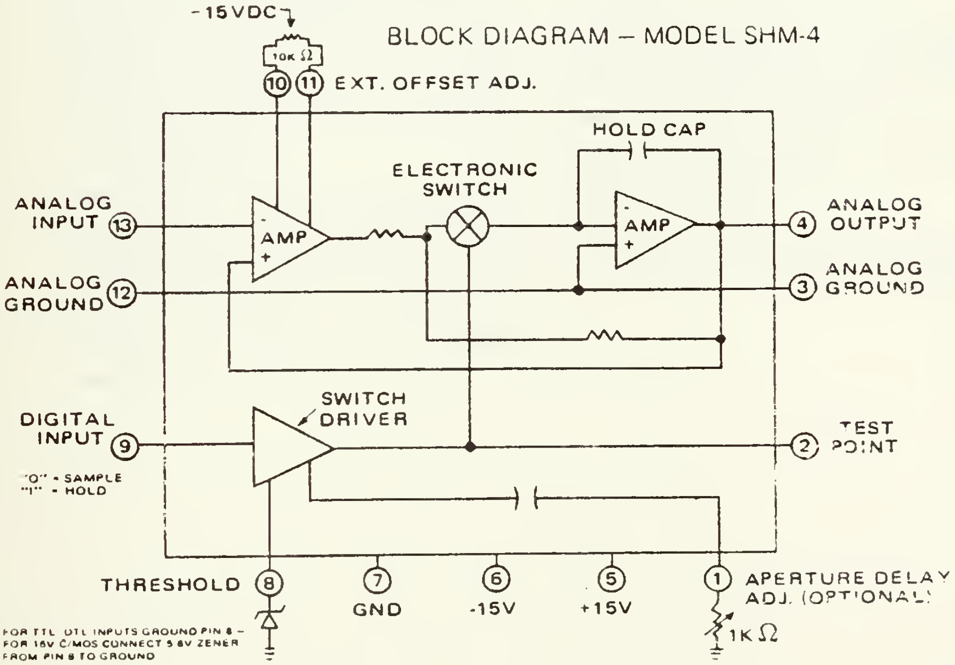
MODEL SHM-4

FOR SIMULTANEOUS SAMPLE AND HOLD APPLICATIONS

FEATURES

□ Fast Acquisition Time	6 µsec
□ Low Droop	20 µV/msec
□ Adjustable Aperture Delay	To zero between units
□ Low Gain Error	± .005%
□ High Input Impedance	100 MΩ

BLOCK DIAGRAM – MODEL SHM-4



DESCRIPTION

The SHM-4 is ideally suited to simultaneous sample and hold applications, where the gain and aperture delay between units must be matched, and where the output droop of the sampled signal is minimized for time shared A/D conversion.

A double inversion circuit in the SHM-4 places the FET sampling switch near ground, which means that all variations of hold step and of aperture delay with input voltage are eliminated.

A unique closed loop design gives high accuracy and allows the rate error[1] to be factory nulled. Rate error is the delay by which the output lags an input ramp and may be expressed in nsec or in mV/V/μsec. For conventional sample and hold applications rate error is not serious because it merely causes an advance in the effective time of hold and tends to cancel out part of the aperture delay. However, for simultaneous applications the aperture delay minus the rate error must be matched between units so that the effective time of hold is the same for all. The SHM-4 accomplishes this by nulling the rate error to less than 1 nanosecond and for critical applications, by providing an external 5 nanosecond adjustment of aperture delay. Also, the high accuracy and low droop of the SHM-4 make it useful in conventional sample and hold applications.

Careful attention to circuit detail in eliminating leakage currents has decreased the output droop to less than 20 microvolts per millisecond allowing several SHM-4 modules to be time shared between one A/D converter.

Datel's Model MM-8 is a complete eight channel solid state analog multiplexer designed for applications which require fast output settling and high transfer accuracy.

The entire multiplexer is self contained in a plastic module measuring 0.8 cubic inches. It contains eight MOS-FET switches with associated driver circuits, each having a current limiter pull-up FET to provide minimum propagation delay, also included is all the necessary decoding logic to enable random channel addressing with a four bit parallel binary input. Two MM-8 multiplexers can be cascaded to provide up to sixteen channels under command from one 4-BIT address. The addressing logic inputs are compatible with DTL/TTL logic levels.

Full scale inputs can be either ±5V or ±10V with a transfer accuracy (input to output) of ±0.01%, provided the output load is a minimum of 10 megohms. The high impedance amplifier provided with Datel's ADC-E, ADC-L and ADC-M series analog/digital converters and SHM Series sample/hold's are quite suitable for this application.

Output settling time for each channel is one microsecond to ±0.01% of full scale and each channel can sequentially switch at a 500 KHz rate. The output of the eight channels is divided into two parallel groups of four.

As stated before, MM-8 is complete and requires only +5VDC, +15VDC and -15VDC (-20VDC) for operation.

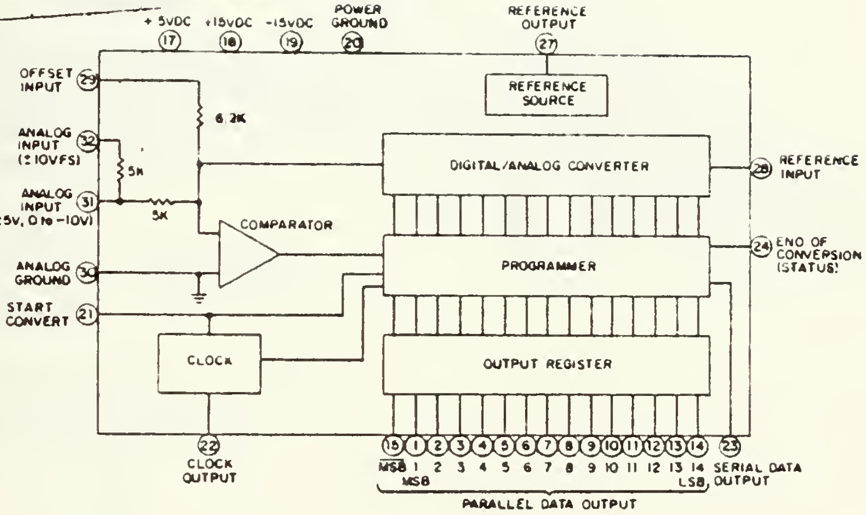
MM-8 modules are 2"L x 1"W x 0.375"H in size come fully encapsulated, and feature dual in line pinning (0.100" grid pin spacing)

[1] Dynamic Accuracy of Sample and Hold Circuits, Datel Systems, Inc., Application Note V1—1.

HIGH RESOLUTION
ANALOG-TO-DIGITAL CONVERTER

MODEL ADC-149

Datel Systems' products are manufactured to high standards of workmanship and are carefully tested to ensure reliable operation. Our delivered products have experienced exceptionally low failure rates. In the event that you experience a malfunction with this product, first carefully check the connections and calibration of the device against the enclosed data sheet. If the problem persists and appears to be an internal malfunction, call the nearest Datel Sales Office for instructions on how to return the unit. Returns must be authorized by the factory and must be shipped prepaid.



GENERAL DESCRIPTION

The ADC-149 is a 14 bit successive approximation type analog to digital converter for OEM use. It was specifically designed to give high resolution and accuracy at moderate cost for incorporation into precision instruments for process control systems and test and measurement systems.

The ADC-149 can resolve 1 part in 16,384 giving an operating dynamic range of 84.3dB. On the 10 volt full scale range it can detect an input change of less than 1 millivolt. Accuracy is adjustable to ±.005% of full scale ±½LSB. The temperature coefficient is held to a low ±15ppm/°C over the 0° to 70°C operating temperature range.

This converter accepts either unipolar or bipolar input voltages of 0 to −10V, 0 to −20V, ±5V, or ±10V full scale by external pin connection and performs a 14 bit conversion in 50 µsec. Several output codes are available including straight binary for unipolar inputs and either offset binary or two's complement for bipolar inputs. Two's complement is obtained by using the $\overline{\text{MSB}}$ output pin. Reverse coding sense is used with the most negative analog input corresponding to full scale digital output. A serial data output is also provided and has a nonreturn-to-zero (NRZ) format. Logic outputs are DTL/TTL compatible and will drive 6 standard TTL loads

ADL PIN CONNECTIONS

A/D Card Code...

22 Pin plug

Pin No.	Ident.	
1	NC	
2	NC	
3	NC	
4	NC	
5	Ch. 0	Analog Voltage Inputs
6	Ch. 1	
7	Ch. 2	
8	Ch. 3	
9	Ch. 4	
10	Ch. 5	
11	Ch. 6	
12	Ch. 7	
13	NC	
14	Relay plug (spare)	
15	Relay plug (Alpha Relay)	
16	Relay plug (Alpha Relay)	
17	Relay plug (Gnd.)	
18	NC	
19	+5 VDC	
20	+15 VDC	Pwr. Supply
21	Gnd.	Buss line
22	-15 VDC	

Output (Data) 16 pin socket

1	A/D Bit 14 (LSB)
2	" " 13
3	" " 12
4	" " 11
5	" " 10
6	" " 9
7	" " 8
8	" " 7
9	" " 6
10	" " 5
11	" " 4
12	" " 3
13	" " 2
14	" " 1 (MSB)
15	A/D E.O.C.
16	Spare

From P3 on Prolog Rack
Input (Command) 16 pin socket

Pin No.	Ident.
1	MUX Ch. 001
2	MUX Ch. 010
3	MUX Ch. 100
4	S/H Command
5	A/D Start
6	Alpha Relay (Incr.)
7	Alpha Relay (Decr.)
8	Relay Plug (spare)
9 to 16	Not used

Relay Plug

A to	P14	Spare Relay
B	P17	Gnd. (Logic)
C	P15	alpha relay (incr.)
D	P16	alpha relay (decr.)
E	P21	Gnd. (Buss)

ADL PIN CONNECTIONS

PROLOG SYSTEM 44 Pin Output Plug (On Top of Card Rack)

Pin No.	CODE Ident.			
1	Out 1-8	to	P3-8	Relay Plug (spare)
2	" 1-7		P3-7	Alpha Relay (decr.)
3	" 1-6		P3-6	Alpha Relay (incr.)
4	" 1-5		P3-5	A/D Start
5	" 1-4		P3-4	S/H Command
6	" 1-3		P3-3	Analog MUX Ch. 100
7	" 1-2		P3-2	" MUX Ch. 010
8	" 1-1		P3-1	" MUX Ch. 001
9	NC			
10	Out 3-8		P4-15	DP (Dec. Pt., Lite Chip 03)
11	" 3-7		P4-14	DP (Lite Chip 02)
12	" 3-6		P4-13	DP (Lite Chip 01)
13	" 3-5		P4-12	DP (lite Chip 00)
14	" 3-4		P4-8	BCD Data 8 (to Lites)
15	" 3-3		P4-7	BCD Data 4
16	" 3-2		P4-6	BCD Data 2
17	" 3-1		P4-5	BCD Data 1
18	NC			
19	"			
20	"			
21	"			
22	"			
A	Out 0-8			
B	" 0-7			
C	" 0-6			
D	" 0-5			
E	" 0-4			
F	" 0-3			
H	" 0-2		TTY Card	JX-12/Jx-14
J	" 0-1		TTY Card	JX-11/JX-9
K	NC			
L	Out 2-8			
M	" 2-7			
N	" 2-6	to	P4-10	(+/- Lite)
P	" 2-5		P4-9	Lite Enable
R	" 2-4		P4-4	Lite MUX 1000
S	" 2-3		P4-3	Lite MUX 0100
T	" 2-2		P4-2	Lite MUX 0010
U	" 2-1		P4-1	Lite MUX 0001
V	NC			
W	"			
X	"			
Y	"			
Z	"			

ADL PIN CONNECTIONS

PROLOG SYSTEM 44 Pin INPUT Plug (On Top of Card Rack)

CODE......

Pin No.	Ident.		
1	In 1-8	to	P1-8, A/D Bit 7
2	" 1-7		P1-7, " " 8
3	" 1-6		P1-6, " " 9
4	" 1-5		P1-5, " " 10
5	" 1-4		P1-4, " " 11
6	" 1-3		P1-3, " " 12
7	" 1-2		P1-2, " " 13
8	" 1-1		P1-1, " " 14 (LSB)
9	NC		
10	In 3-8		NC
11	" 3-7		NC
12	" 3-6		NC
13	" 3-5	to	P2-5, Kyb'd Flag
14	" 3-4		P2-4, Kyb'd (1000)
15	" 3-3		P2-3, " (0100)
16	" 3-2		P2-2, " (0010)
17	" 3-1		P2-1, " (0001)
18	NC		
19	"		
20	"		
21	"		
22	"		
A	In 0-8		
B	" 0-7		
C	" 0-6		
D	" 0-5		
E	" 0-4		
F	" 0-3		
H	" 0-2		
J	" 0-1	to	TTY Card (JX-17)
K	NC		
L	In 2-8	to	P1-16 NC
M	" 2-7		P1-15, A/D EOC (end of conversion)
N	" 2-6		P1-14, A/D Bit 1 (MSB)
P	" 2-5		P1-13, " " 2
R	" 2-4		P1-12, " " 3
S	" 2-3		P1-11, " " 4
T	" 2-2		P1-10, " " 5
U	" 2-1		P1-09, " " 6
V	NC		
W	"		
X	"		
Y	"		
Z	"		

APPENDIX C

MATHEMATICS PACKAGE

Floating point (F.P.) binary numbers are used internally for most internal arithmetic functions. The method is fully explained in the following excerpts from the INTEL Users Library [3].

8008 BINARY FLOATING POINT SYSTEM

ARITHMETIC AND UTILITY PACKAGE

THE ARITHMETIC AND UTILITY SUBROUTINE PACKAGE OF THE 8008 BINARY FLOATING POINT SYSTEM CONTAINS SUBROUTINES FOR PERFORMING THE BASIC ARITHMETIC AND UTILITY OPERATIONS AVAILABLE IN THE SYSTEM.

THE ARITHMETIC AND UTILITY PACKAGE IS CONTAINED IN 768 CONSECUTIVE WORDS OF MEMORY(3 BANKS OF ROM) AND DOES NOT REQUIRE THAT ANY OTHER SOFTWARE BE PRESENT IN MEMORY. THIS PACKAGE USES THE FIRST 54 WORDS OF A BANK OF RAM AS SCRATCHPAD MEMORY.

THE INDIVIDUAL SUBROUTINES INCLUDED IN THE ARITHMETIC AND UTILITY PACKAGE OF THE FLOATING POINT SYSTEM ARE DESCRIBED IN DETAIL BELOW.

8008 BINARY FLOATING POINT SYSTEM

THE 8008 BINARY FLOATING POINT SYSTEM CONSISTS OF A SET OF SUBROUTINES DESIGNED TO PERFORM OPERATIONS ON NUMERIC QUANTITIES REPRESENTED IN A SPECIFIC NOTATION. SUBROUTINES ARE PROVIDED TO PERFORM A VARIETY OF ARITHMETIC AND RELATED OPERATIONS.

THE SUBROUTINES ARE DESIGNED TO BE STORED AND EXECUTED IN READ-ONLY-MEMORY(ROM) AND REQUIRE THE FIRST PORTION OF A BANK OF READ-WRITE-MEMORY(RAM) FOR SCRATCHPAD MEMORY. THE SUBROUTINES ARE SEPARATED INTO A NUMBER OF PACKAGES, EACH CONTAINING SUBROUTINES FOR A GROUP OF RELATED OPERATIONS. THE AMOUNT OF MEMORY(ROM AND RAM) REQUIRED FOR INSTALLATION OF THE SYSTEM IS DEPENDENT UPON THE COMBINATION OF PACKAGES TO BE USED. SCRATCHPAD MEMORY IS INITIALIZED BY A UTILITY SUBROUTINE WHICH MUST BE EXECUTED BEFORE OTHER SUBROUTINES ARE EXECUTED THE FIRST TIME.

IN GENERAL, THE SUBROUTINES HAVE SIMILIAR ENTRY AND EXIT CONDITIONS. UNLESS SPECIFIED DIFFERENTLY IN THE DESCRIPTION OF A SPECIFIC SUBROUTINE, THE SUBROUTINES HAVE THE FOLLOWING CHARACTERISTICS.

SUBROUTINES REQUIRING ONE OPERAND TAKE IT FROM AN INTERNAL FLOATING POINT ACCUMULATOR. SUBROUTINES REQUIRING TWO OPERANDS TAKE ONE FROM THE ACCUMULATOR AND THE OTHER FROM THE MEMORY LOCATION INDICATED BY THE CONTENTS OF THE H AND L REGISTERS UPON ENTRY. THE NUMERIC RESULT OF EACH OPERATION IS STORED IN THE ACCUMULATOR AND IS RETURNED TO THE CALLER IN THE A, B, C, AND D REGISTERS. .

UPON EXIT FROM THE ARITHMETIC SUBROUTINES. THE PROPERTIES OF THE RESULT ARE INDICATED BY THE SETTINGS OF THE CONTROL BITS.

CARRY BIT = 1 — THE RESULT EXCEEDS THE CAPACITY OF THE ACCUMULATOR. THE OTHER CONTROL BITS, THE CONTENTS OF THE HARDWARE REGISTERS, AND THE CONTENTS OF THE ACCUMULATOR ARE MEANINGLESS. THIS SITUATION IS ALSO INDICATED BY A NON-ZERO QUANTITY BEING STORED IN A FLAG WORD.

CARRY BIT = 0 — THE RESULT IS IN RANGE. THE ZERO AND SIGN BITS ARE PROPERLY SET, AND THE A, B, C, AND D REGISTERS CONTAIN A REPRESENTATION OF THE VALUE IN THE ACCUMULATOR.

ZERO BIT = 1 — THE RESULT OF THE OPERATION IS ZERO OR A QUANTITY TOO SMALL TO BE REPRESENTED.

ZERO BIT = 0 — THE RESULT IS NON-ZERO.

SIGN BIT = 1 — THE RESULT IS NEGATIVE.

SIGN BIT = 0 — THE RESULT IS POSITIVE.

DATA ARE REPRESENTED IN A NOTATION WHICH RECORDS EIGHT BITS OF EXPONENT, ONE BIT OF SIGN, AND TWENTY FOUR BITS OF FRACTION. THE LARGEST MAGNITUDE THAT CAN BE REPRESENTED IS APPROXIMATELY 3.6 * 10 ** 38. THE SMALLEST NON-ZERO MAGNITUDE IS APPROXIMATELY 2.7 * 10 ** -39. THE RESOLUTION OF THE NOTATION IS APPROXIMATELY 6.2 * 10 ** -8, I.E., BETTER THAN SEVEN DECIMAL DIGIT PRECISION.

DATA VALUES ARE REPRESENTED IN FOUR CONSECUTIVE MEMORY WORDS WHICH MUST BE IN THE SAME BANK OF MEMORY. THE INTERPRETATION OF THESE WORDS IS SHOWN BELOW.

WORD 1 — IF NON-ZERO, THIS WORD CONTAINS THE EXPONENT PLUS A BIAS OF 200 OCTAL. THE EXPONENT INDICATES THE POWER OF 2 BY WHICH THE FRACTION IS MULTIPLIED TO OBTAIN THE REPRESENTED VALUE. IF THIS WORD IS ZERO THE REPRESENTED VALUE IS ZERO AND WORDS 2, 3, AND 4 ARE MEANINGLESS.

WORD 2, BIT 7 — THIS BIT INDICATES THE SIGN OF THE VALUE: 0 IF POSITIVE, 1 IF NEGATIVE.

WORD 2, BITS 6-0 — THESE BITS PLUS AN ASSUMED 1 IN BIT 7 ARE THE MOST SIGNIFICANT BITS OF THE FRACTION. THE FRACTION IS STORED IN ABSOLUTE FORM (UNSIGNED) WITH THE RADIX POINT POSITIONED TO THE LEFT OF BIT 7. THE VALUE OF THE FRACTION IS THUS LESS THAN 1.0 AND EQUAL TO OR GREATER THAN 0.5.

WORD 3 — THIS WORD CONTAINS THE SECOND MOST SIGNIFICANT EIGHT BITS OF THE FRACTION.

WORD 4 — THIS WORD CONTAINS THE LEAST SIGNIFICANT EIGHT BITS OF THE FRACTION.

EXAMPLES OF DATA NOTATION.

VALUE	WORD1	WORD2	WORD3	WORD4
0.0	000	XXX	XXX	XXX
+1.0	201	000	000	000
-1.0	201	200	000	000
+0.1	175	114	314	314
-100.1	207	310	063	063

X = DONT CARE

FLOATING POINT ACCUMULATOR.

THE FLOATING POINT ACCUMULATOR CONSISTS OF 5 SCRATCHPAD WORDS CONTAINING RESPECTIVELY THE ACCUMULATOR EXPONENT, THE ACCUMULATOR SIGN, AND THREE WORDS OF ACCUMULATOR FRACTION. THE EXPONENT IS RECORDED WITH A BIAS OF 200 OCTAL. AN EXPONENT WORD OF ZERO INDICATES THAT THE VALUE IN THE ACCUMULATOR IS ZERO AND THE REMAINING WORDS OF THE ACCUMULATOR ARE MEANINGLESS. THE SIGN WORD HOLDS 000 IF THE ACCUMULATOR IS NEGATIVE, 200 OCTAL IF POSITIVE. THE FRACTION IS RECORDED AS A NORMALIZED POSITIVE VALUE WITH THE RADIX POINT TO THE LEFT OF THE MOST SIGNIFICANT BIT OF THE FIRST FRACTION WORD.

OVERFLOW FLAG.

THE OVERFLOW FLAG WORD IS PROVIDED AS A CONVENIENCE TO THE USER OF THE FLOATING POINT SYSTEM. THE WORD IS INITIALLY SET TO ZERO AND MAY BE RESET TO ZERO BY THE USER AT ANY TIME. WHEN ANY OF THE SYSTEM SUBROUTINES DETECT AN OVERFLOW CONDITION THE OVERFLOW FLAG IS SET NON-ZERO. THUS THE USER MAY CLEAR THE FLAG, PERFORM A SEQUENCE OF FLOATING POINT OPERATIONS, AND CHECK THE FLAG TO DETERMINE IF AN OVERFLOW OCCURRED ANYWHERE IN THE SEQUENCE.

8008 BINARY FLOATING POINT SYSTEM

THE 8008 BINARY FLOATING POINT SYSTEM CONSISTS OF A SET OF SUBROUTINES DESIGNED TO PERFORM ARITHMETIC OPERATIONS ON NUMERIC QUANTITIES REPRESENTED IN MEMORY.

EACH NUMERIC QUANTITY OCCUPIES FOUR CONSECUTIVE WORDS (32 BITS) OF MEMORY. THE LARGEST MAGNITUDE THAT CAN BE REPRESENTED IS APPROXIMATELY 3.6 TIMES TEN TO THE 38TH POWER. THE SMALLEST NON-ZERO MAGNITUDE THAT CAN BE REPRESENTED IS APPROXIMATELY 2.7 TIMES TEN TO THE MINUS 39TH POWER. EACH NUMERIC QUANTITY IS REPRESENTED WITH A PRECISION OF ONE PART IN APPROXIMATELY 16,000,000.

THE SOFTWARE CONSTITUTING THE FLOATING POINT SYSTEM IS DIVIDED INTO TWO SECTIONS, EACH OF WHICH OCCUPIES 3 BANKS OF ROM OR RAM. SECTION 1 IS INDEPENDENT OF OTHER SOFTWARE. SECTION 2 IS OPERABLE ONLY WHEN SECTION 1 IS AVAILABLE IN MEMORY. IN ADDITION TO MEMORY REQUIRED FOR PROGRAM, 63 WORDS OF RAM ARE USED AS SCRATCHPAD.

SOFTWARE SECTION 1 CONTAINS THE FOLLOWING SUBROUTINES:

LOD - LOAD SPECIFIED DATA INTO THE FLOATING POINT ACCUMULATOR.

ADD - ADD SPECIFIED DATA TO THE FLOATING POINT ACCUMULATOR.

SUB - SUBTRACT SPECIFIED DATA FROM THE FLOATING POINT ACCUMULATOR.

MUL - MULTIPLY SPECIFIED DATA TIMES THE FLOATING POINT ACCUMULATOR.

DIV - DIVIDE SPECIFIED DATA INTO THE FLOATING POINT ACCUMULATOR.

TST - SET CONTROL BITS TO INDICATE ATTRIBUTES OF THE FLOATING POINT ACCUMULATOR.

CHS - CHANGE THE SIGN OF THE FLOATING POINT ACCUMULATOR.

ABS - SET THE SIGN OF THE FLOATING POINT ACCUMULATOR POSITIVE.

STR - STORE IN SPECIFIED MEMORY THE VALUE IN THE REGISTERS AS RETURNED BY OTHER SUBROUTINES.

INIT - MOVE CODE FROM ROM TO RAM IN PREPARATION FOR EXECUTION OF THE MUL AND DIV SUBROUTINES.

SOFTWARE SECTION 2 CONTAINS SUBROUTINES WHICH ARE USED TO CONVERT DATA BETWEEN THE BINARY FLOATING POINT FORMAT AND A DECIMAL FORMAT SUITABLE FOR ENTRY OR DISPLAY ON INPUT/OUTPUT EQUIPMENT. THE DECIMAL FORMAT IS STORED IN MEMORY AS A SERIES OF CHARACTERS. RELATIVELY SIMPLE INPUT/OUTPUT ROUTINES MAY BE USED TO INTERFACE THE MEMORY-RESIDENT CHARACTER STRINGS WITH ANY TYPE OF PHYSICAL I/O DEVICE.

THE CHARACTER STRINGS CONSIST OF BCD REPRESENTATIONS OF DECIMAL DIGITS AND ARBITRARY REPRESENTATIONS OF +, -, ., AN EXPONENTIAL SIGN(LETTER E), AND SPACE. CHARACTER STRINGS MAY NOT CROSS MEMORY BANK BOUNDARIES. AN INPUT STRING IS THEREFORE LIMITED TO 256 CHARACTERS. AN OUTPUT STRING CONSISTS OF 13 CHARACTERS.

THE OUT SUBROUTINE GENERATES CHARACTER STRINGS IN 2 FORMATS; THE CHOICE OF FORMAT DEPENDS ON THE MAGNITUDE OF THE VALUE REPRESENTED.

MAGNITUDES BETWEEN .1000000 AND 9999999. ARE REPRESENTED BY A SPACE OR MINUS SIGN, SEVEN DECIMAL DIGITS AND AN APPROPRIATELY POSITIONED DECIMAL POINT, AND FOUR SPACES.

MAGNITUDES OUTSIDE THE RANGE ARE REPRESENTED BY A SPACE OR MINUS SIGN, A VALUE BETWEEN 1.000000 AND 9.999999, AN EXPONENTIAL SIGN, AND A SIGNED TWO-DIGIT POWER OF TEN.

THE INP SUBROUTINE CONVERTS CHARACTER STRINGS IN EITHER OF THE ABOVE FORMATS, OR A MODIFIED VERSION OF THEM. THE LEADING SIGN MAY BE INCLUDED OR OMITTED. ANY NUMBER OF DIGITS MAY BE USED TO INDICATE THE VALUE, WITH OR WITHOUT AN INCLUDED DECIMAL POINT. IF A POWER-OF-TEN MULTIPLIER IS INDICATED IT MAY BE SIGNED OR UNSIGNED AND MAY CONTAIN ONE OR TWO DIGITS. AN INPUT STRING IS TERMINATED BY THE FIRST CHARACTER WHICH DEPARTS FROM THE FORMAT.

THE FOLLOWING ARE EXAMPLES OF INPUT AND CORRESPONDING OUTPUT CHARACTER STRINGS.

```
3.141593               3.141593
-.0000000000001       -1.000000E-13
+1.6E5                 160000.0
123456789              1.234568E+08
54321E-10              5.432100E-06
-2718281828E-9        -2.718282
```

8008 BINARY FLOATING POINT SYSTEM

FORMAT CONVERSION PACKAGE

THE FORMAT CONVERSION PACKAGE OF THE 8008 BINARY FLOATING POINT SYSTEM CONTAINS SUBROUTINES FOR THE CONVERSION OF DATA BETWEEN THE FLOATING POINT SYSTEM NOTATION AND TWO OTHER FORMATS. THE NON-FLOATING-POINT FORMATS ARE FOUR WORD FIXED POINT FORMAT AND VARIABLE LENGTH CHARACTER STRING FORMAT.

THE FORMAT CONVERSION PACKAGE IS CONTAINED IN 512 CONSECUTIVE WORDS OF MEMORY(2 BANKS OF ROM) AND REQUIRES FOR ITS EXECUTION THAT THE ARITHMETIC AND UTILITY PACKAGE BE AVAILABLE IN MEMORY. THE COMBINATION OF THIS PACKAGE AND THE ARITHMETIC AND UTILITY PACKAGE USES THE FIRST 64 WORDS OF A BANK OF RAM AS SCRATCHPAD MEMORY.

THE FIXED POINT FORMAT DATA PROCESSED BY THIS PACKAGE CONSIST OF 32 BIT BINARY NUMBERS OCCUPYING FOUR WORDS. TWOS COMPLEMENT NOTATION IS USED TO REPRESENT NEGATIVE VALUES.
THE POSITION OF THE BINARY POINT RELATIVE TO THE BITS REPRESENTING THE VALUE IS DENOTED BY A BINARY SCALING FACTOR. THE BINARY SCALING FACTOR IS NOT NORMALLY RECORDED IN THE COMPUTER, BUT WHEN A FORMAT CONVERSION SUBROUTINE IS CALLED THE BINARY SCALING FACTOR MUST BE SPECIFIED(IN THE E REGISTER). A BINARY SCALING FACTOR OF ZERO INDICATES THE BINARY POINT IS IMMEDIATELY TO THE LEFT OF THE MOST SIGNIFICANT OF THE 32 BITS REPRESENTING THE VALUE. A BINARY SCALING FACTOR OF 32 INDICATES THE BINARY POINT IS IMMEDIATELY TO THE RIGHT OF THE LEAST SIGNIFICANT BIT. THE PERMISSIBLE RANGE OF THE BINARY SCALING FACTOR IS -128(200 OCTAL) TO +127(177 OCTAL).

THE CHARACTER STRING FORMAT DATA PROCESSED BY THIS PACKAGE CONSIST OF BINARY REPRESENTATIONS OF CHARACTERS OCCUPYING CONSECUTIVE WORDS OF MEMORY. A CHARACTER STRING MAY NOT CROSS A MEMORY BANK BOUNDARY. THE CHARACTERS WHICH MAY BE INCLUDED IN A CHARACTER STRING, AND THE CORRESPONDING OCTAL REPRESENTATIONS ARE LISTED BELOW.

DECIMAL DIGITS	000B-011B BCD DIGITS
SPACE	360B
+	373B PLUS
-	375B MINUS
.	376B DECIMAL POINT
EXPONENTIAL SIGN	025B LETTER E

(THESE OCTAL REPRESENTATIONS CAN BE CONVERTED TO THE CORRESPONDING ASCII CHARACTERS BY ADDING 060B TO EACH)

THE OUT SUBROUTINE GENERATES CHARACTER STRINGS IN TWO FORMATS, EACH CONSISTING OF 13 CHARACTERS. THE FORMAT USED IN A SPECIFIC CASE IS DEPENDENT UPON THE MAGNITUDE OF THE VALUE REPRESENTED.

SIGNIFICANCE INDEX

THE FLOATING POINT ADD AND SUBTRACT SUBROUTINES RETURN A SIGNIFICANCE INDEX TO THE USER WHEN THE RESULT OF THE OPERATION IS NOT ZERO. THIS INDEX GIVES AN INDICATION OF THE CHANGE IN THE VALUE OF THE ACCUMULATOR EXPONENT AS A RESULT OF THE ARITHMETIC OPERATION PERFORMED. IT IS USED PRIMARILY FOR COMPARISON OF TWO VALUES WHICH ARE EXPECTED TO BE EQUAL, BUT WHICH MAY DIFFER BY A SMALL AMOUNT DUE TO MEASUREMENT OR ROUND-OFF ERRORS. AS AN EXAMPLE, A SIGNIFICANCE INDEX OF 354 OCTAL (-20 DECIMAL) INDICATES THAT THE RESULT OF THE OPERATION IS SMALLER THAN THE OPERANDS BY A FACTOR OF APPROXIMATELY ONE MILLION (2 ** 20).
THE FLOATING POINT TEST, COMPLEMENT AND ABSOLUTE SUBROUTINES RETURN THE SIGNIFICANCE INDEX FROM AN IMMEDIATELY PRECEEDING ADD OR SUBTRACT OPERATION.

HEXADECIMAL NOTATION [4]

Hexadecimal Notation is a convenient way of representing all sixteen combinations of four bits of information with a single character. The most popular character set for displaying Hexadecimal data are the characters 0 thru 9 to represent the binary combinations 0 thru 9 and A B C D E and F to represent the binary combinations 10 thru 15.

Hexadecimal Characters	Binary Bits 8 4 2 1	Decimal Characters
0	0 0 0 0	0
1	0 0 0 1	1
2	0 0 1 0	2
3	0 0 1 1	3
4	0 1 0 0	4
5	0 1 0 1	5
6	0 1 1 0	6
7	0 1 1 1	7
8	1 0 0 0	8
9	1 0 0 1	9
A	1 0 1 0	10
B	1 0 1 1	11
C	1 1 0 0	12
D	1 1 0 1	13
E	1 1 1 0	14
F	1 1 1 1	15

As an extension of this technique, all 256 combinations of 8 bits can be represented by two hexadecimal characters as shown in the following examples.

Hexadecimal Characters	Binary Bits	Decimal Characters
00	0000 0000	0
01	0000 0001	1
3E	0011 1110	52
42	0100 0010	66
E1	1110 0001	225
FF	1111 1111	255

Going further, all 4096 combinations of 12 bits can be represented by three Hexadecimal characters. This technique can be extended indefinitely, adding a Hexadecimal character for each four bits of information.

LIST OF REFERENCES

1. Russell, R. W., A Design Study for a Center Plate Mount for a Wind-tunnel Model, M. S. Thesis,Naval Postgraduate School, Monterey, California, 1977.

2. Englehardt C. D., Data Acquisition System for Unsteady Aerodynamic Investigation, M. S. Thesis,Naval Postgraduate School, Monterey, California, 1977.

3. Intel Corporation, MCS User's Library, p. 8-7, Intel, 1972.

4. Biewer, M, The Designers Guide To Programmed Logic, p. 1-1 to 6-0, Prolog Corporation, 1975.

INITIAL DISTRIBUTION LIST

		No. Copies
1.	Defense Documentation Center Cameron Station Alexandria, Virginia 22314	2
2.	Library, Code 0142 Naval Postgraduate School Monterey, California 93940	2
3.	Department Chairman, Code 67 Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
4.	CDR David Caswell, USN, Code 67Cl Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
5.	Professor L. V. Schmidt, Code 67Sx Department of Aeronautics Naval Postgraduate School Monterey, California 93940	1
6.	LT John David Casko, USN 8 Meyers Street Putnam, Connecticut 06260	1

Thesis 171709
C2749 Casko
c.1 A microprocessor controlled automatic data logging system (ADL).